Temporal Vertex Cover with a Sliding Time Window*

Eleni C. Akrida

Department of Computer Science, University of Liverpool, UK
Eleni.Akrida@liverpool.ac.uk
https://orcid.org/0000-0002-1126-1623

George B. Mertzios

Department of Computer Science, Durham University, UK
George.Mertzios@durham.ac.uk
https://orcid.org/0000-0001-7182-585X

Paul G. Spirakis

Department of Computer Science, University of Liverpool, UK
Department of Computer Engineering & Informatics, University of Patras, Greece
P.Spirakis@liverpool.ac.uk
https://orcid.org/0000-0001-5396-3749

Viktor Zamaraev

Department of Computer Science, Durham University, UK
Viktor.Zamaraev@durham.ac.uk
https://orcid.org/0000-0001-5755-4141

Abstract

Modern, inherently dynamic systems are usually characterized by a network structure, i.e. an underlying graph topology, which is subject to discrete changes over time. Given a static underlying graph G, a temporal graph can be represented via an assignment of a set of integer time-labels to every edge of G, indicating the discrete time steps when this edge is active. While most of the recent theoretical research on temporal graphs has focused on the notion of a temporal path and other "path-related" temporal notions, only few attempts have been made to investigate "non-path" temporal graph problems. In this paper, motivated by applications in sensor and in transportation networks, we introduce and study two natural temporal extensions of the classical problem Vertex Cover. In our first problem, Temporal Vertex Cover, the aim is to cover every edge at least once during the lifetime of the temporal graph, where an edge can only be covered by one of its endpoints at a time step when it is active. In our second, more pragmatic variation Sliding Window Temporal Vertex Cover, we are also given a natural number Δ, and our aim is to cover every edge at *least once* at *every* Δ *consecutive* time steps. In both cases we wish to minimize the total number of "vertex appearances" that are needed to cover the whole graph. We present a thorough investigation of the computational complexity and approximability of these two temporal covering problems. In particular, we provide strong hardness results, complemented by various approximation and exact algorithms. Some of our algorithms are polynomial-time, while others are asymptotically almost optimal under the Exponential Time Hypothesis (ETH) and other plausible complexity assumptions.

2012 ACM Subject Classification Mathematics of computing → Graph theory • Mathematics of computing → Graph algorithms.

* This work was partially supported by the NeST initiative of the School of EEE and CS at the University of Liverpool and by the EPSRC Grants EP/P020372/1 and EP/P02002X/1.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Don Sannella; Article No. 467; pp. 467:1–467:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases Temporal networks, temporal vertex cover, APX-hard, approximation algorithm, Exponential Time Hypothesis.

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.467

Related Version https://arxiv.org/abs/1802.07103

Funding Partially supported by NeST initiative of the School of EEE and CS at the University of Liverpool and by the EPSRC Grants EP/P020372/1 and EP/P02002X/1.

1 Introduction and Motivation

A great variety of both modern and traditional networks are inherently dynamic, in the sense that their link availability varies over time. Information and communication networks, social networks, transportation networks, and several physical systems are only a few examples of networks that change over time [18, 27]. The common characteristic in all these application areas is that the network structure, i.e. the underlying graph topology, is subject to *discrete changes over time*. In this paper we adopt a simple and natural model for time-varying networks which is given with time-labels on the edges of a graph, while the vertex set remains unchanged. This formalism originates in the foundational work of Kempe et al. [20].

▶ **Definition 1** (temporal graph)**.** A *temporal graph* is a pair (G, λ), where $G = (V, E)$ is an underlying (static) graph and $\lambda : E \to 2^{\mathbb{N}}$ is a *time-labeling* function which assigns to every edge of G a set of discrete-time labels.

For every edge $e \in E$ in the underlying graph G of a temporal graph (G, λ), $\lambda(e)$ denotes the set of time slots at which e is *active* in (G, λ). Due to its vast applicability in many areas, this notion of temporal graphs has been studied from different perspectives under various names such as *time-varying* [1, 14, 29], *evolving* [4, 10, 13], *dynamic* [7, 15], and *graphs over time* [24]; for a recent attempt to integrate existing models, concepts, and results from the distributed computing perspective see the survey papers [5–7] and the references therein. Data analytics on temporal networks have also been very recently studied in the context of summarizing networks that represent sports teams' activity data to discover recurring strategies and understand team tactics [22], as well as extracting patterns from interactions between groups of entities in a social network [21].

Motivated by the fact that, due to causality, information in temporal graphs can "flow" only along sequences of edges whose time-labels are increasing, most temporal graph parameters and optimization problems that have been studied so far are based on the notion of temporal paths and other "path-related" notions, such as temporal analogues of distance, diameter, reachability, exploration, and centrality [2, 3, 12, 25, 26]. In contrast, only few attempts have been made to define "non-path" temporal graph problems. Motivated by the contact patterns among high-school students, Viard et al. [31, 32], and later Himmel et al. [17], introduced and studied Δ-cliques, an extension of the concept of cliques to temporal graphs, in which all vertices interact with each other at least once every Δ consecutive time steps within a given time interval.

In this paper we introduce and study two natural temporal extensions of the problem Vertex Cover in static graphs, which take into account the dynamic nature of the network. In the first and simpler of these extensions, namely Temporal Vertex Cover (for short, TVC), every edge e has to be "covered" at least once during the lifetime T of the network (by one of its endpoints), and this must happen at a time step t when e is active. The goal is then to cover all edges with the minimum total number of such "vertex appearances". On the other hand, in many real-world applications where scalability is important, the lifetime T can be arbitrarily large but the network still needs to remain sufficiently covered. In such cases, as well as in safety-critical systems (e.g. in military applications), it may not be satisfactory enough that an edge is covered just *once* during the *whole lifetime* of the network. Instead, every edge must be covered at least once within *every small* Δ*-window* of time (for an appropriate value of Δ), regardless of how large the lifetime is; this gives rise to our second optimization problem, namely Sliding Window Temporal Vertex Cover (for short, SW-TVC). Formal definitions of our problems TVC and SW-TVC are given in Section 2.

Our two temporal extensions of Vertex Cover are motivated by applications in sensor networks and in transportation networks. In particular, several works in the field of sensor networks considered problems of placing sensors to cover a whole area or multiple critical locations, e.g. for reasons of surveillance. Such studies usually wish to minimize the number of sensors used or the total energy required [11, 16, 23, 28, 33]. Our temporal vertex cover notions are an abstract way to economically meet such covering demands as time progresses.

To further motivate the questions raised in this work, consider a network whose links represent transporting facilities which are not always available, while the availability schedule per link is known in advance. We wish to check each transporting facility and certify "OK" at least once per facility during every (reasonably small) window of time. It is natural to assume that the checking is done in the presence of an inspecting agent at an endpoint of the link (i.e. on a vertex), since such vertices usually are junctions with local offices. The agent can inspect more than one link at the same day, provided that these links share this vertex and that they are all alive (i.e. operating) at that day. Notice that the above is indeed an application drawn from real-life, as regular checks in roads and trucks are paramount for the correct operation of the transporting sector, according to both the European Commission[1] and the American Public Transportation Association[2].

1.1 Our contribution

In this paper we present a thorough investigation of the complexity and approximability of the problems Temporal Vertex Cover (TVC) and Sliding Window Temporal Vertex Cover (SW-TVC) on temporal graphs. We first prove in Section 3 that Set Cover is equivalent to a special case of TVC on star temporal graphs (i.e. when the underlying graph G is a star), which immediately provides several complexity and algorithmic consequences for TVC. In particular, TVC remains NP-complete even on star temporal graphs, and it does not admit a polynomial-time $(1-\varepsilon)\ln n$-approximation algorithm, unless NP has $n^{O(\log\log n)}$-time deterministic algorithms. On the positive side, TVC on star temporal graphs with n vertices can be $(H_{n-1}-\frac{1}{2})$-approximated in polynomial time, where $H_n=\sum_{i=1}^{n}\frac{1}{i}\approx\ln n$ is the nth harmonic number. Similar equivalence with Hitting Set yields that for any $\varepsilon<1$, TVC on star temporal graphs cannot be optimally solved in $O(2^{\varepsilon n})$ time, assuming the Strong Exponential Time Hypothesis (SETH). We complement these results by showing that TVC

[1] According to the European Commission (see `https://ec.europa.eu/transport/road_safety/topics/vehicles/inspection_en`), "roadworthiness checks (such as on-the-spot roadside inspections and periodic checks) not only make sure your vehicle is working properly, they are also important for environmental reasons and for ensuring fair competition in the transport sector".

[2] According to the American Public Transportation Association (see `http://www.apta.com/resources/standards/Documents/APTA-RT-VIM-RP-019-03.pdf` "developing minimum inspection, maintenance, testing and alignment procedures maintains rail transit trucks in a safe and reliable operating condition".

on general temporal graphs admits a polynomial-time randomized approximation algorithm with expected ratio $O(\ln n)$.

In Section 4 and in the reminder of the paper we deal with our second problem, SW-TVC. We prove in Section 4.1 a strong complexity lower bound on arbitrary temporal graphs. More specifically we prove that, for *any* (arbitrarily growing) functions $f : \mathbb{N} \to \mathbb{N}$ and $g : \mathbb{N} \to \mathbb{N}$, there exists a constant $\varepsilon \in (0, 1)$ such that SW-TVC cannot be solved in $f(T) \cdot 2^{\varepsilon n \cdot g(\Delta)}$ time, assuming the Exponential Time Hypothesis (ETH). This ETH-based lower bound turns out to be asymptotically almost tight, as we present an exact dynamic programming algorithm with running time $O(T\Delta(n+m) \cdot 2^{n(\Delta+1)})$. This worst-case running time can be significantly improved in certain special temporal graph classes. In particular, when the "snapshot" of (G, λ) at every time step has vertex cover number bounded by k, the running time becomes $O(T\Delta(n + m) \cdot n^{k(\Delta+1)})$. That is, when Δ is a constant, this algorithm is polynomial in the input size on temporal graphs with bounded vertex cover number at every time step. Notably, when every snapshot is a star (i.e. a superclass of the star temporal graphs studied in Section 3) the running time of the algorithm is $O(T\Delta(n + m) \cdot 2^{\Delta})$.

In Section 5 we prove strong inapproximability results for SW-TVC even when restricted to temporal graphs with length $\Delta = 2$ of the sliding window. In particular, we prove that this problem is APX-hard (and thus does not admit a *Polynomial Time Approximation Scheme (PTAS)*, unless P = NP), even when $\Delta = 2$, the maximum degree in the underlying graph G is at most 3, and every connected component at every graph snapshot has at most 7 vertices. Finally, in Section 6 we provide a series of approximation algorithms for the general SW-TVC problem, with respect to various incomparable temporal graph parameters. In particular, we provide polynomial-time approximation algorithms with approximation ratios (i) $O(\ln n + \ln \Delta)$, (ii) $2k$, where k is the maximum number of times that each edge can appear in a sliding Δ time window (thus implying a ratio of 2Δ in the general case), (iii) d, where d is the maximum vertex degree at every snapshot of (G, λ). Note that, for $d = 1$, the latter result implies that SW-TVC can be optimally solved in polynomial time whenever every snapshot of (G, λ) is a matching.

2 Preliminaries and notation

A theorem proving that a problem is NP-hard does not provide much information about how efficiently (although not polynomially, unless P = NP) this problem can be solved. In order to prove some useful complexity lower bounds, we mostly need to rely on some complexity hypothesis that is stronger than"P $\neq$ NP". The *Exponential Time Hypothesis (ETH)* is one of the established and most well-known such complexity hypotheses.

▶ Exponential Time Hypothesis (ETH [19]). There exists an $\varepsilon < 1$ such that 3SAT cannot be solved in $O(2^{\varepsilon n})$ time, where n is the number of variables in the input 3-CNF formula.

Given a (static) graph G, we denote by $V(G)$ and $E(G)$ the sets of its vertices and edges, respectively. An edge between two vertices u and v of G is denoted by uv, and in this case u and v are said to be *adjacent* in G. The maximum label assigned by λ to an edge of G, called the *lifetime* of (G, λ), is denoted by $T(G, \lambda)$, or simply by T when no confusion arises. That is, $T(G, \lambda) = \max\{t \in \lambda(e) : e \in E\}$. For every $i, j \in \mathbb{N}$, where $i \leq j$, we denote $[i, j] = \{i, i + 1, \dots, j\}$. Throughout the paper we consider temporal graphs with *finite lifetime* T, and we refer to each integer $t \in [1, T]$ as a *time slot* of (G, λ). The *instance* (or *snapshot*) of (G, λ) *at time* t is the static graph $G_t = (V, E_t)$, where $E_t = \{e \in E : t \in \lambda(e)\}$. For every $i, j \in [1, T]$, where $i \leq j$, we denote by $(G, \lambda)|_{[i,j]}$ the restriction of (G, λ) to the time slots $i, i + 1, \dots, j$, i.e. $(G, \lambda)|_{[i,j]}$ is the sequence of the instances $G_i, G_{i+1}, \dots, G_j$. We assume in the remainder of the paper that every edge of G appears in at least one time slot until T, namely $\bigcup_{t=1}^{T} E_t = E$.

Although some optimization problems on temporal graphs may be hard to solve in the worst case, an optimal solution may be efficiently computable when the input temporal graph (G, λ) has special properties, i.e. if (G, λ) belongs to a special *temporal graph class* (or *time-varying graph class* [5, 7]). To specify a temporal graph class we can restrict (a) the *underlying topology* G, or (b) the *time-labeling* λ, i.e. the temporal pattern in which the time-labels appear, or both.

▶ **Definition 2.** Let (G, λ) be a temporal graph and let $\mathcal{X}$ be a class of (static) graphs. If $G \in \mathcal{X}$ then (G, λ) is an $\mathcal{X}$ *temporal graph.* On the other hand, if $G_i \in \mathcal{X}$ for every $i \in [1, T]$, then (G, λ) is an *always* $\mathcal{X}$ *temporal graph.*

In the remainder of the paper we denote by $n = |V|$ and $m = |E|$ the number of vertices and edges of the underlying graph G, respectively, unless otherwise stated. Furthermore, unless otherwise stated, we assume that the labeling λ is arbitrary, i.e. (G, λ) is given with an explicit list of labels for every edge. That is, the *size* of the input temporal graph (G, λ) is $O\left(|V| + \sum_{t=1}^{T} |E_t|\right) = O(n + mT)$. In other cases, where λ is more restricted, e.g. if λ is periodic or follows another specific temporal pattern, there may exist more succinct representations of the input temporal graph.

For every $u \in V$ and every time slot t, we denote the *appearance of vertex u at time t* by the pair (u, t). That is, every vertex u has T different appearances (one for each time slot) during the lifetime of (G, λ). Similarly, for every vertex subset $S \subseteq V$ and every time slot t we denote the *appearance of set S at time t* by (S, t). With a slight abuse of notation, we write $(S, t) = \bigcup_{v \in S}(v, t)$. A *temporal vertex subset* of (G, λ) is a set $\mathcal{S} \subseteq \{(v, t) : v \in V, 1 \leq t \leq T\}$ of vertex appearances in (G, λ). Given a temporal vertex subset $\mathcal{S}$, for every time slot $t \in [1, T]$ we denote by $\mathcal{S}_t = \{(v, t) : (v, t) \in \mathcal{S}\}$ the set of all vertex appearances in $\mathcal{S}$ at the time slot t. Similarly, for any pair of time slots $i, j \in [1, T]$, where $i \leq j$, $\mathcal{S}|_{[i,j]}$ is the restriction of the vertex appearances of $\mathcal{S}$ within the time slots $i, i + 1, \ldots, j$. Note that the *cardinality* of the temporal vertex subset $\mathcal{S}$ is $|\mathcal{S}| = \sum_{1 \leq t \leq T} |\mathcal{S}_t|$.

2.1 Temporal Vertex Cover

Let $\mathcal{S}$ be a temporal vertex subset of (G, λ). Let $e = uv \in E$ be an edge of the underlying graph G and let (w, t) be a vertex appearance in $\mathcal{S}$. We say that vertex w *covers* the edge e if $w \in \{u, v\}$, i.e. w is an endpoint of e; in that case, edge e is *covered* by vertex w. Furthermore we say that the vertex appearance (w, t) *temporally covers* the edge e if (i) w covers e and (ii) $t \in \lambda(e)$, i.e. the edge e is *active* during the time slot t; in that case, edge e is *temporally covered* by the vertex appearance (w, t). We now introduce the notion of a *temporal vertex cover* and the optimization problem Temporal Vertex Cover.

▶ **Definition 3.** Let (G, λ) be a temporal graph. A *temporal vertex cover* of (G, λ) is a temporal vertex subset $\mathcal{S} \subseteq \{(v, t) : v \in V, 1 \leq t \leq T\}$ of (G, λ) such that every edge $e \in E$ is *temporally covered* by at least one vertex appearance (w, t) in $\mathcal{S}$.

Temporal Vertex Cover (TVC)
Input: A temporal graph (G, λ).
Output: A temporal vertex cover $\mathcal{S}$ of (G, λ) with the smallest cardinality $|\mathcal{S}|$.

Note that TVC is a natural temporal extension of the problem Vertex Cover on static graphs. In fact, Vertex Cover is the special case of TVC where $T = 1$. Thus TVC is clearly NP-complete, as it also trivially belongs to NP.

2.2 Sliding Window Temporal Vertex Cover

In the notion of a temporal vertex cover given in Section 2.1, the requirement is that every edge is temporally covered at least once during the lifetime T of the input temporal graph (G, λ). On the other hand, in many real-world applications where scalability is important, the lifetime T can be arbitrarily large. In such cases it may not be satisfactory enough that an edge is temporally covered just *once* during the whole lifetime of the temporal graph. Instead, in such cases it makes sense that every edge is temporally covered by some vertex appearance at least once during *every small period* Δ of time, regardless of how large the lifetime T is. Motivated by this, we introduce in this section a natural *sliding window* variant of the TVC problem, which offers a greater scalability of the solution concept.

For every time slot $t \in [1, T - \Delta + 1]$, we define the *time window* $W_t = [t, t + \Delta - 1]$ as the sequence of the Δ consecutive time slots $t, t + 1, \ldots, t + \Delta - 1$. Furthermore we denote by $E[W_t] = \bigcup_{i \in W_t} E_i$ the union of all edges appearing at least once in the time window W_t. Finally we denote by $\mathcal{S}[W_t] = \{(v, i) \in \mathcal{S} : i \in W_t\}$ the restriction of the temporal vertex subset $\mathcal{S}$ to the window W_t. We are now ready to introduce the notion of a *sliding Δ-window temporal vertex cover* and the optimization problem Sliding Window Temporal Vertex Cover.

▶ **Definition 4.** Let (G, λ) be a temporal graph with lifetime T and let $\Delta \leq T$. A *sliding Δ-window temporal vertex cover* of (G, λ) is a temporal vertex subset $\mathcal{S} \subseteq \{(v, t) : v \in V, 1 \leq t \leq T\}$ of (G, λ) such that, for every time window W_t and for every edge $e \in E[W_t]$, e is *temporally covered* by at least one vertex appearance (w, t) in $\mathcal{S}[W_t]$.

> Sliding Window Temporal Vertex Cover (SW-TVC)
>
> **Input:** A temporal graph (G, λ) with lifetime T, and an integer $\Delta \leq T$.
> **Output:** A sliding Δ-window temporal vertex cover $\mathcal{S}$ of (G, λ) with the smallest cardinality $|\mathcal{S}|$.

Whenever the parameter Δ is a fixed constant, we will refer to the above problem as the Δ-TVC (i.e. Δ is now a part of the problem name). Note that the problem TVC defined in Section 2.1 is the special case of SW-TVC where $\Delta = T$, i.e. where there is only one Δ-window in the whole temporal graph. Another special case[3] of SW-TVC is the problem 1-TVC, whose optimum solution is obtained by iteratively solving the (static) problem Vertex Cover on each of the T static instances of (G, λ); thus 1-TVC fails to fully capture the time dimension in temporal graphs.

3 Hardness and approximability of TVC

In this section we investigate the complexity of Temporal Vertex Cover (TVC). First we prove in Section 3.1 that TVC on star temporal graphs is equivalent to both Set Cover and Hitting Set, and derive several complexity and algorithmic consequences for TVC.

[3] The problem 1-TVC has already been investigated under the name “evolving vertex cover” in the context of maintenance algorithms in dynamic graphs [8]; similar “evolving” variations of other graph covering problems have also been considered, e.g. the “evolving dominating set” [6].

In Section 3.2 we use randomized rounding technique to prove that TVC on general temporal graphs admits a polynomial-time randomized approximation algorithm with expected ratio $O(\ln n)$. This result is complemented by our results in Section 6.1 where we prove that SW-TVC (and thus also TVC) can be deterministically approximated with ratio $H_{2n\Delta} - \frac{1}{2} \approx \ln n + \ln 2\Delta - \frac{1}{2}$ in polynomial time.

3.1 Hardness on star temporal graphs

In the next theorem we reduce Set Cover to TVC on star temporal graphs, and vice versa. Our hardness results are complemented in Theorem 6 by reducing from Hitting Set.

▶ **Theorem 5.** TVC *on star temporal graphs is NP-complete and it admits a polynomial-time* $(H_{n-1} - \frac{1}{2})$*-approximation algorithm. Furthermore, for any* $\varepsilon > 0$*,* TVC *on star temporal graphs does not admit any polynomial-time* $(1-\varepsilon)\ln n$*-approximation algorithm, unless NP has* $n^{O(\log\log n)}$*-time deterministic algorithms.*

▶ **Theorem 6.** *For every* $\varepsilon < 1$*,* TVC *on star temporal graphs cannot be optimally solved in* $O(2^{\varepsilon n})$ *time, unless the Strong Exponential Time Hypothesis (SETH) fails.*

3.2 A randomized rounding algorithm for TVC

In this section we provide a linear programming relaxation of TVC, and then, with the help of a randomized rounding technique, we construct a feasible solution whose expected size is within a factor of $O(\ln n)$ of the optimal size.

▶ **Theorem 7.** *There exists a polynomial-time randomized approximation algorithm for* TVC *with expected approximation factor* $O(\ln n)$*.*

4 An almost tight algorithm for SW-TVC

In this section we investigate the complexity of Sliding Window Temporal Vertex Cover (SW-TVC). First we prove in Section 4.1 a strong lower bound on the complexity of optimally solving this problem on arbitrary temporal graphs. More specifically we prove that, for *any* (arbitrarily growing) functions $f : \mathbb{N} \to \mathbb{N}$ and $g : \mathbb{N} \to \mathbb{N}$, there exists a constant $\varepsilon \in (0, 1)$ such that SW-TVC cannot be solved in $f(T) \cdot 2^{\varepsilon n \cdot g(\Delta)}$ time, assuming the Exponential Time Hypothesis (ETH). This ETH-based lower bound turns out to be asymptotically almost tight. In fact, we present in Section 4.2 an exact dynamic programming algorithm for SW-TVC whose running time on an arbitrary temporal graph is $O(T\Delta(n+m) \cdot 2^{n(\Delta+1)})$, which is asymptotically almost optimal, assuming ETH. In Section 4.3 we prove that our algorithm can be refined so that, when the vertex cover number of each snapshot G_i is bounded by a constant k, the running time becomes $O(T\Delta(n + m) \cdot n^{k(\Delta+1)})$. That is, when Δ is a constant, this algorithm is polynomial in the input size on temporal graphs with bounded vertex cover number at every slot. Notably, for the class of always star temporal graphs (i.e. a superclass of the star temporal graphs studied in Section 3.1) the running time of the algorithm is $O(T\Delta(n + m) \cdot 2^{\Delta})$.

4.1 A complexity lower bound

In the the following theorem we prove a strong ETH-based lower bound for SW-TVC. This lower bound is asymptotically almost tight, as we present in Section 4.2 a dynamic programming algorithm for SW-TVC with running time $O(T\Delta(n + m) \cdot 2^{n\Delta})$, where n and m are the numbers of vertices and edges in the underlying graph G, respectively.

▶ **Theorem 8.** *For* any *two (arbitrarily growing) functions $f, g : \mathbb{N} \to \mathbb{N}$, there exists a constant $\varepsilon \in (0,1)$ such that* SW-TVC *cannot be solved in $f(T) \cdot 2^{\varepsilon n \cdot g(\Delta)}$ time assuming ETH, where n is the number of vertices in the underlying graph G of the temporal graph.*

4.2 An exact dynamic programming algorithm

The main idea of our dynamic programming algorithm for SW-TVC is to scan the temporal graph from left to right with respect to time (i.e. to scan the snapshots G_i increasingly on i), and at every time slot to consider all possibilities for the vertex appearances at the previous Δ time slots. Let (G, λ) be a temporal graph with n vertices and lifetime T, and let $\Delta \leq T$. For every $t = 1, 2, \ldots, T - \Delta + 1$ and every Δ-tuple of vertex subsets $A_1, \ldots, A_\Delta$ of G, we define $f(t; A_1, A_2, \ldots, A_\Delta)$ to be the smallest cardinality of a sliding Δ-window temporal vertex cover $\mathcal{S}$ of $(G, \lambda)|_{[1,t+\Delta-1]}$, such that $\mathcal{S}_t = (A_1, t)$, $\mathcal{S}_{t+1} = (A_2, t{+}1)$, $\ldots$, $\mathcal{S}_{t+\Delta-1} = (A_\Delta, t{+}\Delta{-}1)$. If there exists no sliding Δ-window temporal vertex cover $\mathcal{S}$ of $(G, \lambda)|_{[1,t+\Delta-1]}$ with these prescribed vertex appearances in the time slots $t, t+1, \ldots, t+\Delta-1$, then we define $f(t; A_1, A_2, \ldots, A_\Delta) = \infty$. Note that, once we have computed all possible values of the function $f(\cdot)$, then the optimum solution of SW-TVC on (G, λ) has cardinality

$$\mathrm{OPT}_{\text{SW-TVC}}(G, \lambda) = \min_{A_1, A_2, \ldots, A_\Delta \subseteq V} \{f(T - \Delta + 1; A_1, A_2, \ldots, A_\Delta)\}. \quad (1)$$

▶ **Lemma 9.** *Let (G, λ) be a temporal graph, where $G = (V, E)$. Let $2 \leq t \leq T - \Delta + 1$ and let $A_1, A_2, \ldots A_\Delta$ be a Δ-tuple of vertex subsets of the underlying graph G. Suppose that $\bigcup_{i=1}^{\Delta}(A_i, t + i - 1)$ is a temporal vertex cover of $(G, \lambda)|_{[t,t+\Delta-1]}$. Then*

$$f(t; A_1, A_2, \ldots, A_\Delta) = |A_\Delta| + \min_{X \subseteq V} \{f(t - 1; X, A_1, \ldots, A_{\Delta-1})\}. \quad (2)$$

Using the recursive computation of Lemma 9, we are now ready to present Algorithm 1 for computing the value of an optimal solution of SW-TVC on a given arbitrary temporal graph (G, λ). Note that Algorithm 1 can be easily modified such that it also computes the actual optimum solution of SW-TVC (instead of only its optimum cardinality). The proof of correctness and running time analysis of Algorithm 1 are given in the next theorem.

Algorithm 1 SW-TVC

Input: A temporal graph (G, λ) with lifetime T, where $G = (V, E)$, and a natural $\Delta \leq T$.
Output: The smallest cardinality of a sliding Δ-window temporal vertex cover in (G, λ).

```
for t = 1 to T − Δ + 1 do
    for all A_1, A_2, ..., A_Δ ⊆ V do
        if ⋃_{i=1}^{Δ} (A_i, t + i − 1) is a temporal vertex cover of (G, λ)|_[t,t+Δ−1] then
            if t = 1 then
                f(t; A_1, A_2, ..., A_Δ) ← Σ_{i=1}^{Δ} |A_i|
            else
                f(t; A_1, A_2, ..., A_Δ) ← |A_Δ| + min_{X⊆V} {f(t − 1; X, A_1, ..., A_{Δ−1})}
        else
            f(t; A_1, A_2, ..., A_Δ) ← ∞
return min_{A_1,...,A_Δ⊆V} {f(T − Δ + 1; A_1, ..., A_Δ)}
```

▶ **Theorem 10.** *Let (G, λ) be a temporal graph, where $G = (V, E)$ has n vertices and m edges. Let T be its lifetime and let Δ be the length of the sliding window. Algorithm 1 computes in $O(T\Delta(n + m) \cdot 2^{n(\Delta+1)})$ time the value of an optimal solution of* SW-TVC *on (G, λ).*

4.3 Always bounded vertex cover number temporal graphs

Let k be a constant and let $\mathcal{C}_k$ be the class of graphs with the vertex cover number at most k. The next theorem follows now from the analysis of Theorem 10.

▶ **Theorem 11.** SW-TVC *on always $\mathcal{C}_k$ temporal graphs can be solved in $O(T\Delta(n+m)\cdot n^{k(\Delta+1)})$ time.*

In particular, in the special, yet interesting, case of always star temporal graphs, our search at every step reduces to just one binary choice for each of the previous Δ time slots, of whether to include the central vertex of a star in a snapshot or not. Hence we have the following theorem as a direct implication of Theorem 11.

▶ **Theorem 12.** SW-TVC *on always star temporal graphs can be solved in $O(T\Delta(n+m)\cdot 2^{\Delta})$ time.*

5 Approximation hardness of 2-TVC

In this section we study the complexity of Δ-TVC where Δ is constant. We start with an intuitive observation that, for every fixed Δ, the problem $(\Delta+1)$-TVC is at least as hard as Δ-TVC. Indeed, let $\mathcal{A}$ be an algorithm that computes a minimum-cardinality sliding $(\Delta+1)$-window temporal vertex cover of (G,λ). It is easy to see that a minimum-cardinality sliding Δ-window temporal vertex cover of (G,λ) can also be computed using $\mathcal{A}$, if we amend the input temporal graph by inserting one edgeless snapshot after every Δ consecutive snapshots of (G,λ).

Since the 1-TVC problem is equivalent to solving T instances of Vertex Cover (on static graphs), the above reduction demonstrates in particular that, for any natural Δ, Δ-TVC is at least as hard as Vertex Cover. Therefore, if Vertex Cover is hard for a class $\mathcal{X}$ of static graphs, then Δ-TVC is also hard for the class of always $\mathcal{X}$ temporal graphs. In this section, we show that the converse is not true. Namely, we reveal a class $\mathcal{X}$ of graphs, for which Vertex Cover can be solved in *linear* time, but 2-TVC is NP-hard on always $\mathcal{X}$ temporal graphs. In fact, we show the even stronger result that 2-TVC is APX-hard (and thus does not admit a PTAS, unless P = NP) on always $\mathcal{X}$ temporal graphs.

To prove the main result (in Theorem 14) we start with an auxiliary lemma, showing that Vertex Cover is APX-hard on the class $\mathcal{Y}$ of graphs which can be obtained from a cubic graph by subdividing every edge exactly 4 times.

▶ **Lemma 13.** Vertex Cover *is APX-hard on $\mathcal{Y}$.*

Let now $\mathcal{X}$ be the class of graphs whose connected components are induced subgraphs of the graph obtained from the star with three leaves by subdividing each of its edges exactly once. Clearly, Vertex Cover is linearly solvable on graphs from $\mathcal{X}$. We will show that 2-TVC is APX-hard on always $\mathcal{X}$ temporal graphs by using a reduction from Vertex Cover on $\mathcal{Y}$.

▶ **Theorem 14.** 2-TVC *is APX-hard on always $\mathcal{X}$ temporal graphs.*

Proof. To prove the theorem we will reduce Vertex Cover on $\mathcal{Y}$ to 2-TVC on always $\mathcal{X}$ temporal graphs. Let $H=(V,E)$ be a graph in $\mathcal{Y}$. First we will show how to construct an always $\mathcal{X}$ temporal graph (G,λ) of lifetime 2. Then we will prove that the size τ of a minimum vertex cover of H is equal to the size σ of a minimum-cardinality sliding 2-window temporal vertex cover of (G,λ).

Let $R \subseteq V$ be the set of vertices of degree 3 in H. We define (G, λ) to be a temporal graph of lifetime 2, where snapshot G_1 is obtained from H by removing the edges with both ends being at distance exactly 2 from R, and snapshot $G_2 = H - R$. Figure 1 illustrates the reduction for $H = K_4$.

Let $\mathcal{S} = (S_1, 1) \cup (S_2, 2)$ be an arbitrary sliding 2-window temporal vertex cover of (G, λ) for some $S_1, S_2 \subseteq V$. Since every edge of H belongs to at least one of the graphs G_1 and G_2, the set $S_1 \cup S_2$ covers all the edges of H. Hence, $\tau \leq |S_1 \cup S_2| \leq |S_1| + |S_2| = |\mathcal{S}|$. As $\mathcal{S}$ was chosen arbitrarily we further conclude that $\tau \leq \sigma$.

To show the converse inequality, let $C \subseteq V$ be a minimum vertex cover of H. Let S_1 be those vertices in C which either have degree 3, or have a neighbor of degree 3. Let also $S_2 = C \setminus S_1$. We claim that $(S_1, 1) \cup (S_2, 2)$ is a sliding 2-window temporal vertex cover of (G, λ). First, let $e \in E$ be an edge in H incident to a vertex of degree 3. Then, by the construction, e is active only in time slot 1, i.e. $e \in E_1 \setminus E_2$, and a vertex v in C covering e belongs to S_1. Hence, e is temporally covered by $(v, 1)$ in (G, λ). Let now $e \in E$ be an edge in H whose both end vertices have degree 2. If one of the end vertices of e is adjacent to a vertex of degree 3 in H, then, by the construction, e is active in both time slots 1 and 2. Therefore, since $C = S_1 \cup S_2$, edge e will be temporally covered in (G, λ) in at least one of the time slots. Finally, if none of the end vertices of e is adjacent to a vertex of degree 3 in H, then e is active only in time slot 2, i.e. $e \in E_2 \setminus E_1$. Moreover, by the construction a vertex v in C covering e belongs to S_2. Hence, e is temporally covered by $(v, 2)$ in (G, λ). This shows that $(S_1, 1) \cup (S_2, 2)$ is a sliding 2-window temporal vertex cover of (G, λ), and therefore $\sigma \leq |S_1| + |S_2| = |C| = \tau$.

Note that the size of a minimum vertex cover of H is equal to the size of a minimum-cardinality sliding 2-window temporal vertex cover of (G, λ) and that any feasible solution to 2-TVC on (G, λ) of size r defines a vertex cover of H of size at most r. Thus, since Vertex Cover is APX-hard on $\mathcal{Y}$ by Lemma 13 and the reduction is approximation-preserving, it follows that 2-TVC is APX-hard as well. ◀

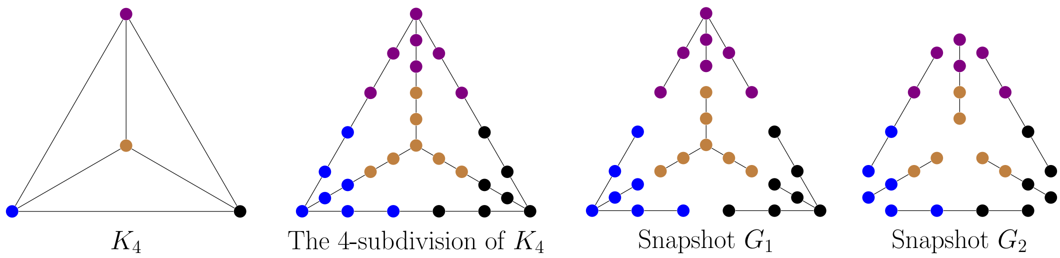

Figure 1 A cubic graph K_4, its 4-subdivision, and the corresponding snapshots G_1 and G_2

6 Approximation algorithms

In this section we provide several approximation algorithms for SW-TVC with respect to different temporal graph parameters. As the various approximation factors that are achieved are incomparable, the best option for approximating an optimal solution depends on the specific application domain and the specific values of those parameters.

6.1 Approximations in terms of T, Δ, and the largest edge frequency

We begin by presenting a reduction from SW-TVC to Set Cover, which proves useful for deriving approximation algorithms for the original problem. Consider an instance,

(G,λ) and $\Delta \le T$, of the SW-TVC problem. Construct an instance of SET COVER as follows: Let the universe be $U = \{(e,t) : e \in E[W_t], t \in [1, T-\Delta+1]\}$, i.e. the set of all pairs (e,t) of an edge e and a time slot t such that e appears (and so must be temporally covered) within window W_t. For every vertex appearance (v,s) we define $C_{v,s}$ to be the set of elements (e,t) in the universe U, such that (v,s) temporally covers e in the window W_t. Formally, $C_{v,s} = \{(e,t) : v \text{ is an endpoint of } e, e \in E_s, \text{ and } s \in W_t\}$. Let $\mathcal{C}$ be the family of all sets $C_{v,s}$, where $v \in V, s \in [1,T]$. The following lemma shows that finding a minimum-cardinality sliding Δ-window temporal vertex cover of (G,λ) is equivalent to finding a minimum-cardinality family of sets $C_{v,s}$ that covers the universe U.

▶ **Lemma 15.** *A family $\mathcal{C} = \{C_{v_1,t_1}, \dots, C_{v_k,t_k}\}$ is a set cover of U if and only if $\mathcal{S} = \{(v_1,t_1), \dots, (v_k,t_k)\}$ is a sliding Δ-window temporal vertex cover of (G,λ).*

$O(\ln n + \ln \Delta)$-approximation. In the instance of SET COVER constructed by the above reduction, every set $C_{v,s}$ in $\mathcal{C}$ contains at most $n\Delta$ elements of the universe U. Indeed, the vertex appearance (v,s) temporally covers at most $n-1$ edges, each in at most Δ windows (namely from window $W_{s-\Delta+1}$ up to window W_s). Thus we can apply the polynomial-time greedy algorithm from [9] for SET COVER which achieves an approximation ratio of $H_{n\Delta} - \frac{1}{2} = \sum_{i=1}^{n\Delta} \frac{1}{i} - \frac{1}{2} \approx \ln n + \ln \Delta - \frac{1}{2}$.

$2k$-approximation, where k is the maximum edge frequency. Given a temporal graph (G,λ) and an edge e of G, the Δ-frequency of e is the maximum number of time slots at which e appears within a Δ-window. Let k denote the maximum Δ-frequency over all edges of G. Clearly, for a particular Δ-window W_t, an edge $e \in E[W_t]$ can be temporally covered in W_t by at most $2k$ vertex appearances. So in the above reduction to SET COVER, every element $(e,t) \in U$ belongs to at most $2k$ sets in $\mathcal{C}$. Therefore, the optimal solution of the constructed instance of SET COVER can be approximated within a factor of $2k$ in polynomial time [30], yielding a $2k$-approximation for SW-TVC.

2Δ-approximation. Since the maximum Δ-frequency of an edge is always upper-bounded by Δ, the previous algorithm gives a worst-case polynomial-time 2Δ-approximation for SW-TVC on arbitrary temporal graphs.

6.2 Approximation in terms of maximum degree of snapshots

In this section we give a polynomial-time d-approximation algorithm for the SW-TVC problem on *always degree at most d* temporal graphs, that is, temporal graphs where the maximum degree in each snapshot is at most d. In particular, the algorithm computes an optimum solution (i.e. with approximation ratio $d = 1$) for always matching (i.e. always degree at most 1) temporal graphs. As a building block, we first provide an exact $O(T)$-time algorithm for optimally solving SW-TVC in the class of single-edge temporal graphs, namely temporal graphs whose underlying graph is a single edge.

Single-edge temporal graphs

Consider a temporal graph (G_0,λ) where G_0 is the single-edge graph, i.e. $V(G_0) = \{u,v\}$ and $E(G_0) = \{uv\}$. We reduce SW-TVC on (G_0,λ) to an instance of INTERVAL COVERING.

INTERVAL COVERING
Input: A family $\mathcal{I}$ of intervals in the line.
Output: A minimum-cardinality subfamily $\mathcal{I}' \subseteq \mathcal{I}$ such that $\bigcup_{I \in \mathcal{I}} = \bigcup_{I \in \mathcal{I}'}$.

An easy linear-time greedy algorithm for the Interval Covering picks at each iteration, among the intervals that cover the leftmost uncovered point, the one with largest finishing time. Algorithm 2 implements this simple rule in the context of the SW-TVC problem.

Algorithm 2 SW-TVC on single-edge temporal graphs

Input: A temporal graph (G_0, λ) of lifetime T with $V(G_0) = \{u, v\}$, and $\Delta \leq T$.
Output: A minimum-cardinality sliding Δ-window temporal vertex cover $\mathcal{S}$ of (G_0, λ).

$\mathcal{S} \leftarrow \emptyset$
$t = 1$
while $t \leq T - \Delta + 1$ **do**
  if $\exists r \in [t, t + \Delta - 1]$ such that $uv \in E_t$ **then**
    choose maximum such r and add (u, r) to $\mathcal{S}$
    $t \leftarrow r + 1$
  else
    $t \leftarrow t + 1$
return $\mathcal{S}$

▶ **Lemma 16.** *Algorithm 2 solves* SW-TVC *on a single-edge temporal graph and can be implemented to work in time $O(T)$.*

Always degree at most d temporal graphs

We present now the main algorithm of this section, the idea of which is to independently solve SW-TVC for every possible single-edge temporal subgraph of a given temporal graph by Algorithm 2, and take the union of these solutions. We will show that this algorithm is a d-approximation algorithm for SW-TVC on always degree at most d temporal graphs.

Let (G, λ) be a temporal graph, where $G = (V, E)$, $|V| = n$, and $|E| = m$. For every edge $e = uv \in E$, let $(G[\{u, v\}], \lambda)$ denote the temporal graph where the underlying graph is the induced subgraph $G[\{u, v\}]$ of G and the labels of e are exactly the same as in (G, λ).

Algorithm 3 d-approximation of SW-TVC on always degree at most d temporal graphs

Input: An always degree at most d temporal graph (G, λ) of lifetime T, and $\Delta \leq T$.
Output: A sliding Δ-window temporal vertex cover $\mathcal{S}$ of (G, λ).

for $i = 1$ to T **do**
  $\mathcal{S}_i \leftarrow \emptyset$
for every edge $e = uv \in E(G)$ **do**
  Compute an optimal solution $\mathcal{S}'(uv)$ of the problem for $(G[\{u, v\}], \lambda)$ by Algorithm 2
  for $i = 1$ to T **do**
    $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \mathcal{S}'_i(uv)$
return $\mathcal{S}$

▶ **Lemma 17.** *Algorithm 3 is a $O(mT)$-time d-approximation algorithm for* SW-TVC *on always degree at most d temporal graphs.*

Note that, in the case of always matching temporal graphs, the maximum degree in each snapshot is $d = 1$, so the above d-approximation actually yields an exact algorithm.

▶ **Corollary 18.** SW-TVC *can be optimally solved in $O(mT)$ time on the class of always matching temporal graphs.*

References

1 Eric Aaron, Danny Krizanc, and Elliot Meyerson. DMVP: foremost waypoint coverage of time-varying graphs. In *Proceedings of the 40th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 29–41, 2014.

2 Eleni C. Akrida, Leszek Gasieniec, George B. Mertzios, and Paul G. Spirakis. Ephemeral networks with random availability of links: The case of fast networks. *Journal of Parallel and Distributed Computing*, 87:109–120, 2016.

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

4 Binh-Minh Bui-Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(2):267–285, 2003.

5 Arnaud Casteigts and Paola Flocchini. Deterministic Algorithms in Dynamic Networks: Formal Models and Metrics. Technical report, Defence R&D Canada, April 2013. URL: `https://hal.archives-ouvertes.fr/hal-00865762`.

6 Arnaud Casteigts and Paola Flocchini. Deterministic Algorithms in Dynamic Networks: Problems, Analysis, and Algorithmic Tools. Technical report, Defence R&D Canada, April 2013. URL: `https://hal.archives-ouvertes.fr/hal-00865764`.

7 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems (IJPEDS)*, 27(5):387–408, 2012.

8 Arnaud Casteigts, Bernard Mans, and Luke Mathieson. On the feasibility of maintenance algorithms in dynamic graphs. *CoRR*, abs/1107.2722, 2011. URL: `http://arxiv.org/abs/1107.2722`.

9 Rong chii Duh and Martin Fürer. Approximation of k-set cover by semi-local optimization. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 256–264, 1997.

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

11 Stefan Dobrev, Stephane Durocher, Mohsen Eftekhari Hesari, Konstantinos Georgiou, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil M. Shende, and Jorge Urrutia. Complexity of barrier coverage with relocatable sensors in the plane. *Theoretical Computer Science*, 579:64–73, 2015.

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

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

14 Paola Flocchini, Bernard Mans, and Nicola Santoro. Exploration of periodically varying graphs. In *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC)*, pages 534–543, 2009.

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

16 Mohsen Eftekhari Hesari, Evangelos Kranakis, Danny Krizanc, Oscar Morales Ponce, Lata Narayanan, Jaroslav Opatrny, and Sunil M. Shende. Distributed algorithms for barrier coverage using relocatable sensors. *Distributed Computing*, 29(5):361–376, 2016.

17 Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Adapting the bron-kerbosch algorithm for enumerating maximal cliques in temporal graphs. *Social Network Analysis and Mining*, 7(1):35:1–35:16, 2017.

18 P. Holme and J. Saramäki, editors. *Temporal Networks*. Springer, 2013.

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

20 David Kempe, Jon M. Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC)*, pages 504–513, 2000.

21 Orestis Kostakis and Aristides Gionis. On mining temporal patterns in dynamic graphs, and other unrelated problems. In *Proceedings of the 6th International Conference on Complex Networks and Their Applications (COMPLEX NETWORKS)*, pages 516–527, 2017.

22 Orestis Kostakis, Nikolaj Tatti, and Aristides Gionis. Discovering recurring activity in temporal networks. *Data Mining and Knowledge Discovery*, 31(6):1840–1871, 2017.

23 Evangelos Kranakis, Danny Krizanc, Flaminia L. Luccio, and Brett Smith. Maintaining intruder detection capability in a rectangular domain with sensors. In *Proceedings of the 11th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS)*, pages 27–40, 2015.

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

25 George B. Mertzios, Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis. Temporal network optimization subject to connectivity constraints. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP), Part II*, pages 657–668, 2013.

26 Othon Michail and Paul G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical Computer Science*, 634:1–23, 2016.

27 Othon Michail and Paul G. Spirakis. Elements of the theory of dynamic networks. *Communications of the ACM*, 61(2):72–72, January 2018.

28 Sotiris E. Nikoletseas and Paul G. Spirakis. Probabilistic distributed algorithms for energy efficient routing and tracking in wireless sensor networks. *Algorithms*, 2(1):121–157, 2009. URL: `https://doi.org/10.3390/a2010121`.

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

30 Vijay V. Vazirani. *Approximation algorithms*. Springer, 2003.

31 Jordan Viard, Matthieu Latapy, and Clémence Magnien. Revealing contact patterns among high-school students using maximal cliques in link streams. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1517–1522, 2015.

32 Tiphaine Viard, Matthieu Latapy, and Clémence Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609:245–252, 2016.

33 Chuan Zhu, Chunlin Zheng, Lei Shu, and Guangjie Han. A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, 35(2):619–632, 2012.