


Faster Exploration of Degree-Bounded Temporal Graphs

Thomas Erlebach

Department of Informatics, University of Leicester, Leicester, England


te17@leicester.ac.uk

 <https://orcid.org/0000-0002-4470-5868>

Jakob T. Spooner

Department of Informatics, University of Leicester, Leicester, England

jts21@leicester.ac.uk

 <https://orcid.org/0000-0003-3816-6308>

Abstract

A temporal graph can be viewed as a sequence of static graphs indexed by discrete time steps. The vertex set of each graph in the sequence remains the same; however, the edge sets are allowed to differ. A natural problem on temporal graphs is the TEMPORAL EXPLORATION problem (TEXP): given, as input, a temporal graph \mathcal{G} of order n , we are tasked with computing an *exploration schedule* (i.e., a temporal walk that visits all vertices in \mathcal{G}), such that the time step at which the walk arrives at the last unvisited vertex is minimised (we refer to this time step as the *arrival time*). It can be easily shown that general temporal graphs admit exploration schedules with arrival time no greater than $O(n^2)$. Moreover, it has been shown previously that there exists an infinite family of temporal graphs for which any exploration schedule has arrival time $\Omega(n^2)$, making these bounds tight for general TEXP instances. We consider restricted instances of TEXP, in which the temporal graph given as input is, in every time step, of maximum degree d ; we show an $O(\frac{n^2}{\log n})$ bound on the arrival time when d is constant, and an $O(d \log d \cdot \frac{n^2}{\log n})$ bound when d is given as some function of n .

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases temporal graph exploration, algorithmic graph theory, *NP*-complete problem

Digital Object Identifier 10.4230/LIPIcs.MFCS.2018.36

1 Introduction

A natural generalisation of the static, undirected graph is one by which a notion of time is introduced to the edge set. Such a generalisation provides a flexible tool for the modelling of problems/scenarios of a dynamic nature; particularly those which incorporate some temporal aspect into their structure. Informally, a *temporal graph* can be viewed as a graph whose edges are allowed to change over time. Time, herein, refers to an interval comprised of discrete time steps; the model we consider sees a temporal graph as an ordered sequence of static graphs indexed by the steps in this time interval (we call the number of steps contained in the interval the *lifetime* of the graph). In each graph of the sequence, the edge set may differ, whilst the vertex set remains constant. Additionally, we require that the edges that appear in each time step originate from some pre-specified static graph, which we call the *underlying graph*. It is this potential for the edges connecting the vertices in the graph to change over time that allows us to model problems, scenarios and systems in which the relationships between entities can evolve. Various “dynamic graph” models exist; they, and the problems defined on them, have been explored in a number of other studies. For an



© Thomas Erlebach and Jakob T. Spooner;
licensed under Creative Commons License CC-BY

43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018).

Editors: Igor Potapov, Paul Spirakis, and James Worrell; Article No. 36; pp. 36:1–36:13

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

overview we refer the reader to [9] and [4]. Since the addition of an element of dynamicity fundamentally makes temporal graphs different from their static predecessors, it is clear that the development of new techniques for their analysis (and for the analysis of problems defined upon them) is required. In this paper, we take an algorithmic standpoint, and consider a restricted case of the problem of computing a walk around a temporal graph that visits all vertices by the earliest time step possible.

1.1 Contribution

This paper considers the problem of TEMPORAL EXPLORATION, or TEXP (defined originally in [10] by Michail and Spirakis), which asks that, given a temporal graph \mathcal{G} , we compute an *exploration schedule* (i.e., a walk visiting each vertex in \mathcal{G} at least once) such that the time step at which the last unvisited vertex is reached is minimal amongst all possible schedules (we call this time step the *arrival time*). In [10], Michail and Spirakis gave an $O(dn)$ upper bound on the arrival time of exploration schedules for arbitrary temporal graphs, where d is the *dynamic diameter* of the graph. Erlebach et al. [5] observed that, in general temporal graphs that are connected in every step, d can be at most $n - 1$, implying an $O(n^2)$ bound on arrival times in the worst case. Moreover, they provide an explicit construction of an infinite family of graphs for which any exploration schedule has arrival time $\Omega(n^2)$, making these bounds tight for general instances of TEXP. We note that this construction is, as far as we are aware, the only one that has been shown to require this many steps for exploration; this, coupled with the fact that all graphs within this family contain a vertex of high degree ($n - 1$) in each time step, motivates our examination of the problem when restricted to instances in which the temporal graph is of maximum degree d in every time step. Further, provided in [5] is a construction of a degree-bounded family of graphs (with planar underlying graphs), such that each graph in the family is of maximum degree 2 in every time step. They prove $\Omega(n \log n)$ arrival times of any exploration schedule for any graph in this family. No better bound than the general $O(n^2)$ bound is known for the arrival time of exploration schedules for degree-bounded temporal graphs. We make the first step towards narrowing this large gap by showing that, for any graph contained within the class of degree-bounded temporal graphs, one can guarantee that there exists an exploration schedule, W , such that the arrival time of W is $O(d \log d \cdot \frac{n^2}{\log n})$ when d is given as some function of n , and $O(\frac{n^2}{\log n})$ when d is constant.

1.2 Related work

A number of previous studies have considered how standard results, problems and definitions are affected when viewed in the context of a variety of dynamic graph models. For example, it was found by Berman in [1] that the vertex variant of Menger's theorem does not hold when applied to a particular model of dynamic graphs (termed *scheduled networks*), in which each edge is assigned both an arrival and departure time. On the other hand, Kempe et al. showed in [7] that, under this same model, there is a class of temporal graphs for which the vertex variant of Menger's theorem does hold – they show this by means of a forbidden minor characterisation.

As well as their previously mentioned results, Kempe et al. [7] showed that it is *NP*-complete to decide whether or not there exist two vertex-disjoint, *time-respecting* paths (i.e., each edge in the path is traversed during a time step that is strictly greater than the previous edge) between a given source and sink. Under a similar model, Bui-Xuan et al. [3] consider the problems of computing an *s-t* path in a temporal graph that is *shortest* (minimal number

of edges), *foremost* (arrives at t at the earliest possible time step) or *fastest* (the time spent between traversing the first and last edges in the path is minimal), showing that there are a number of natural path parameters one might wish to optimise when considering dynamic graphs.

In [8], Mertzios et al. consider a temporal graph model in which each edge, e , is labelled with the set of time steps during which e appears in the graph. They specify a polynomial-time algorithm for the problem of computing a foremost path between two given vertices, and, complementing the work in [1] and [7], present a temporal analogue of Menger's theorem which holds for general graphs adhering to their model.

Brodén et al. [2] study a temporal analogue of the TSP problem, in which the graph is a complete graph in every step, and a cost belonging to the set $\{1, 2\}$ is assigned to each edge; additionally, these costs can change in each time step. They assume that the costs of the edges can change at most k times over the course of the graph's lifetime, and manage to provide a polynomial-time approximation algorithm with approximation ratio $2 - \frac{2}{3k}$. In [10], Michail and Spirakis also considered this model, showing the general problem to be *APX*-hard, whilst improving on the results of [2] with a $(1.7 + \varepsilon)$ -approximation algorithm.

Under the same model as [8], Michail and Spirakis [10] formally introduced the problem of *TEMPORAL EXPLORATION*, showing that the general problem of deciding whether a temporal graph admits any valid exploration schedule is *NP*-complete if the graph is not assumed to be connected in each time step. They suggested making the assumption that the graph is connected in every step. Erlebach et al. [5] further studied the *TEXP* problem under this assumption. In addition to their previously mentioned results, they obtained an $O(n^{1-\varepsilon})$ -inapproximability result, for any $\varepsilon > 0$, ruling out the possibility of any constant-factor approximation algorithms. Additionally, they considered the problem of *TEXP* when the input graph is subject to various structural restrictions: amongst other things, they proved $O(n)$ arrival times for temporal graphs in which the underlying graph is a cycle; $O(n^{1.5}k^2 \log n)$ arrival times for underlying graphs of treewidth k ; and $O(n \log^3 n)$ arrival times for underlying graphs that are $2 \times n$ grids. From a different angle, they also considered a model in which the edges of the graph are present in each step with a particular probability, or appear with a certain regularity.

In [6], Fluschnik et al. considered the *NP*-hard problem of *TEMPORAL (s, z)-SEPARATION*, in which we are asked to separate two vertices, s and z , in a given temporal graph, by removing from it a minimal number of vertices. Similarly to [5], they showed that there are a number of restricted graph classes for which *TEMPORAL (s, z)-SEPARATION* becomes easier, and a number for which the problem remains hard to solve; for example they showed that the problem becomes fixed-parameter tractable when parameterised by $k + l$, where k is the size of a solution, and l is the longest temporal path in the input graph. Negatively, they showed that for graphs in which at most one edge is present in every time step, the problem remains *NP*-hard.

2 Preliminaries

In this section, we introduce those definitions key to formulating the general problem of *TEMPORAL EXPLORATION*.

► **Definition 1** (Temporal graph). We represent a temporal graph, \mathcal{G} , with *underlying graph*, $G = (V, E)$, using an ordered sequence of static graphs: $\mathcal{G} = \langle G_1, G_2, \dots, G_\tau \rangle$. Further, we let the set $V(\mathcal{G}) = V$ and $|V| = n$. The subscripts $i \in \{1, 2, \dots, \tau\}$ indexing the graphs in the

sequence are the discrete *time steps* 1 to τ , where τ is known as the *lifetime* of \mathcal{G} . Each G_i represents the structure of \mathcal{G} in time step i . More precisely, $G_i = (V, E_i)$ is a subgraph of G ; in particular, $V(G_i) = V(G)$, $E_i \subseteq E$, for all $1 \leq i \leq \tau$.

► **Definition 2** (Temporal walk). A temporal walk W through a temporal graph \mathcal{G} is given as an alternating sequence of vertices and edge-time pairs,

$$W = v_0, (e_0, i_0), v_1, (e_1, i_1), v_2, \dots, v_{k-1}, (e_{k-1}, i_{k-1}), v_k,$$

that starts at vertex v_0 and ends at vertex v_k . Additionally, we require that $i_0 < i_1 < \dots < i_{k-1}$, so that an agent following W can traverse at most one edge per time step. Each edge-time pair, (e_j, i_j) , denotes the traversal of edge $e_j = \{v_j, v_{j+1}\}$ at timestep i_j . For such a traversal to be possible, e_j must be present in graph G_{i_j} , i.e. $e_j \in E_{i_j}$. We say that a walk, W , *departs* at time $i_0 = \delta(W)$ and *arrives* at time $i_{k-1} + 1 = \alpha(W)$, and may refer to the time steps $\delta(W)$ and $\alpha(W)$ as the *departure* and *arrival* times of W , respectively. Finally, let $t_0 \leq \delta(W)$ be an arbitrary time step, and assume that an agent waits at vertex v_0 for $\delta(W) - t_0$ time steps before traversing edge e_0 ; we refer to the difference, $|W| = \alpha(W) - t_0$, as the *duration* of W .

The following result is implied by Lemma 1 in [5]. It provides an upper bound on the number of time steps it might take to reach one vertex from another in a temporal graph that is connected in every time step.

► **Lemma 3** (Erlebach, Hoffmann and Kammer [5]). *Let \mathcal{G} be a temporal graph with vertex set V , and assume \mathcal{G} is connected in each step. Then an agent situated at any vertex $u \in V$ at any time $t \leq \tau - n$ can reach any other vertex $v \in V$ in at most $|V| - 1 = n - 1$ steps, i.e., by time step $t + n - 1$.*

► **Definition 4** (Exploration schedule). We say that a walk, W , is an *exploration schedule* if, for all $v \in V(\mathcal{G})$, there exists some $v_i \in W$ such that $v = v_i$. Additionally, W is said to be *foremost* if it reaches the n -th unique vertex $v \in V(\mathcal{G})$ at time $\alpha(W)$, and if there exists no other temporal walk W' , such that $\alpha(W') < \alpha(W)$.

► **Definition 5** (Temporal walk concatenation). We define two temporal walks, W and W' , to be *compatible under concatenation* if the departure time of W' is greater than or equal to the arrival time of W (i.e., $\delta(W') \geq \alpha(W)$), and W' departs from the same vertex at which W arrives. The result of their concatenation is the walk obtained by following W , then following W' directly after.

► **Problem** (TEMPORAL EXPLORATION). An instance of the general TEMPORAL EXPLORATION (TEXP) problem is given as a pair (\mathcal{G}, s) , where $\mathcal{G} = \langle G_1, G_2, \dots, G_\tau \rangle$ is an arbitrary temporal graph with lifetime $\tau \geq |V(\mathcal{G})|^2 = n^2$ (in order to ensure that there exist feasible solutions for any instance), and $s \in V(\mathcal{G})$ is a start vertex. The problem then asks for a temporal walk, W , such that W is an exploration schedule, W is foremost, and W departs from vertex s . We make the additional assumption that the graph is connected in each step; without this it could happen that there exists no valid exploration schedule.

We note that since (as part of any instance of the TEXP problem) we are given a temporal graph, $\mathcal{G} = \langle G_1, G_2, \dots, G_\tau \rangle$, any candidate algorithm for the problem knows, in advance, the structure of the graph in each step of its lifetime (and therefore knows the dynamic structure of the temporal graph's edge set in advance). The following section introduces those definitions specific to the proof of our main result.

3 Exploring degree-bounded temporal graphs

We now turn our attention to restricted instances of the TEMPORAL EXPLORATION problem, in which the input graph is of bounded degree in each time step. We proceed by introducing those definitions central to the proof of our main result.

► **Definition 6** (Temporal graph of bounded degree). Let $\mathcal{G}^d = \langle G_1^d, G_2^d, \dots, G_\tau^d \rangle$ be a temporal graph of order $|V(\mathcal{G}^d)| = n$, and *lifetime* τ . It is said that the degree of \mathcal{G}^d is bounded by d if, for all i , $\max_{v \in V(G_i^d)} \deg_{G_i^d}(v) \leq d$. We note that d may be given as a constant, or as some function of n .

(Note that we do not place any restriction on the underlying graph of \mathcal{G}^d .) Consider now a partitioning of the vertex set, $V(\mathcal{G}^d)$, of a degree-bounded temporal graph, \mathcal{G}^d , into distinct parts T and L , such that $T \cup L = V(\mathcal{G}^d)$. We refer to the set T as the *terminal* set, containing $|T| = k$ terminal vertices $u \in T$, and L as the *leftover* set, containing the $|L| = n - k$ leftover vertices $v \in L$. The set T of terminals will contain, initially, the set of all unvisited vertices in \mathcal{G}^d , whilst the set L will contain the specified start vertex, s . Our aim is to form $\Omega(\frac{k}{d})$ disjoint ordered pairs of terminal vertices, (u, u') , such that there is a temporal walk through \mathcal{G}^d that departs from u , arrives at u' , and has duration no longer than $O(\frac{dn}{k})$. In doing so, we obtain a subset $T' \subseteq T$ of terminals which are arrived at by such a walk; by setting $T = T'$, moving all terminals not reachable by such a walk into L , and forming pairs amongst those $u \in T$ again, we obtain a new collection of walks, each of which can be concatenated with exactly one of the walks obtained from the previous application. Repeated application of this process will form the basis of our overall exploration approach, eventually enabling us to explore $\Theta(\log_d n)$ vertices in $O(dn)$ steps. The following definitions will prove useful in showing this – in each of them we consider an arbitrary temporal graph, \mathcal{G} , during a time period starting at time step t , and ending at time step t' :

► **Definition 7** (Reachable set). The reachable set of a vertex, $x \in V(\mathcal{G})$, is the set $R_x(t, t') = \{y \in V(\mathcal{G}) \mid \text{there exists a walk from } x \text{ to } y \text{ starting at time } t \text{ and ending at time } t', \text{ with } t \leq t \leq t' \leq t'\}$. If t and t' are made clear, or are deducible from the context, we simply write R_x , rather than $R_x(t, t')$.

► **Definition 8** (Reachable pair). We call an ordered pair of terminal vertices, $(u, u') \in T \times T$, a *reachable pair* if $u' \in R_u(t, t')$ (with $1 \leq t \leq t' \leq \tau$) and $u \neq u'$. When it is clear from the context, we may simply refer to a reachable pair as a *pair*. We also say that a reachable pair is *formed* in the step $t' - 1$, in which u' is added to $R_u(t, t')$.

► **Definition 9** (Home set). We define a *home set* of a leftover vertex, $v \in L$, to be a set $H_v(t, t')$ of terminal vertices such that, for any terminal vertex u , the condition $u \in H_v(t, t')$ implies the following: $v \in R_u(t, t')$, and u does not already belong to a reachable pair. As such, there exist temporal walks in \mathcal{G} departing from all $u \in H_v(t, t')$ and arriving at v . Again, we write H_v rather than $H_v(t, t')$ when t and t' are clear from the context.

Note that Definition 9 does not define home sets in a unique way. It only requires that $u \in H_v(t, t')$ implies $v \in R_u(t, t')$, but there is no requirement that all vertices u that satisfy $v \in R_u(t, t')$ are included in the home set. We will specify how to construct home sets in a certain way, and the home sets resulting from our construction will have the additional property that they contain at most two vertices. The purpose of home sets and the details of their construction will become clear in the proof of Lemma 13.

► **Definition 10** (Spread of a terminal vertex). We refer to the number of home sets that a particular terminal vertex $u \in T$ belongs to as the *spread* of u .

As previously discussed, we first wish to show that in a degree-bounded temporal graph, \mathcal{G}^d , enough disjoint reachable pairs are formed during any period of $O(\frac{dn}{k})$ time steps (here we take k to be the number of terminals contained in T at the start of the time period we are considering). To achieve this, we introduce the following potential function, which tracks the number of terminals that each leftover vertex has contained within its respective home set. By first showing that we can increase the value of this potential function by a large enough amount in each of the $O(\frac{dn}{k})$ considered steps, we will be able to prove that $\Omega(\frac{k}{d})$ disjoint reachable pairs can be found during those same steps.

► **Definition 11** (Potential function, ϕ). Consider an arbitrary time step t , and let i be the current time step ($1 \leq t \leq i \leq \tau$). Further, let L_0, L_1, L_2 be the sets of leftover vertices $v \in L$, such that $|H_v(t, i)| = 0$, $|H_v(t, i)| = 1$, and $|H_v(t, i)| = 2$, respectively. In other words, L_0, L_1 and L_2 are the sets of leftover vertices that have 0, 1 and 2 terminals contained in their home sets at time i . Clearly, $L = L_0 \cup L_1 \cup L_2$ at any time, since the home sets we consider grow to a size no larger than 2. We denote by P_v^i the *potential value* of vertex v at time i , and define it for all i as follows:

$$P_v^i = \begin{cases} 1, & \text{if } v \in L_0 \text{ (at time } i\text{)} \\ 2, & \text{if } v \in L_1 \text{ (at time } i\text{)} \\ 3, & \text{if } v \in L_2 \text{ (at time } i\text{)}. \end{cases}$$

Given this, we introduce our potential function, ϕ , taking as argument a bounded-degree temporal graph, \mathcal{G}^d :

$$\phi_t^i(\mathcal{G}^d) = \sum_{v \in L} P_v^i,$$

so that $\phi_t^i(\mathcal{G}^d)$ is the sum of the values, P_v^i , during time step i ($t \leq i \leq \tau$), given that we began tracking the value of ϕ at time t . When t is clear from the context, we may refer to $\phi_t^i(\mathcal{G}^d)$ at arbitrary i ($t \leq i \leq \tau$) as $\phi^i(\mathcal{G}^d)$ or, if i is clear from the context, as $\phi(\mathcal{G}^d)$ or ϕ .

► **Observation 12.** If we assume that we begin tracking the value of our potential function from some time t , such that no $v \in L$ belongs to the reachable set of any $u \in T$ at time t , then initially, $\phi^t(\mathcal{G}^d) = |L|$. Further to this, $P_v^i \leq 3$ for any $v \in L$, and $|L| \leq |V(\mathcal{G}^d)|$. Therefore, $\phi^i(\mathcal{G}^d) \leq 3|V(\mathcal{G}^d)| = 3n$ for any $t \leq i \leq \tau$.

We note that when a reachable pair (u, u') is formed, then both $u, u' \in T$ are no longer considered as candidates for further reachable pairs (since we require our pairs to be disjoint). Therefore, we remove u and u' from T and place them into L . As a result, it is possible for the number of leftover vertices to grow over the course of time, allowing for larger potential values in future time steps. Moreover, in order to increase our potential in any given step, we must be able to find terminal vertices that can be added to the home set of a leftover vertex in that step.

By Definition 9, $u \in H_v$ implies that u does not already belong to any reachable pair. It therefore follows that the forming of a pair in \mathcal{G}^d can also cause the potential to drop, since we are required to remove u and u' from all home sets that they are contained within. Whilst some decrease in potential is inevitable, it is important that we ensure that the decrease is not too large in any single step. If the home sets of many leftover vertices contain the same terminal, and that terminal goes on to form a pair, then it will be removed from all of these home sets, possibly generating a large drop in potential. If this type of behaviour occurs too often, it might happen that we are unable to obtain a large enough amount of potential in total, and as a result, not be able to form the required amount of reachable pairs. This issue is fully addressed by Lemma 13 in the following subsection.

3.1 Exploration method

Following the discussion above, all is in place to fully describe our overall approach to computing exploration schedules (in degree-bounded temporal graphs) that are guaranteed to have arrival time $O(d \log d \cdot \frac{n^2}{\log n})$. We begin by showing that it is possible to obtain a potential increase of $\frac{k-2p}{2d}$ in any given time step, where p is the number of pairs that have already been formed between vertices in T . For the following, assume that we have fixed a time step t , and that we consider \mathcal{G}^d from time t onwards.

► **Lemma 13.** *Consider a degree-bounded temporal graph, \mathcal{G}^d , in an arbitrary timestep i , whose vertex set has been partitioned into sets T (terminals) and L (leftovers), which initially contained $|T| = k$ and $|L| = n - k$ vertices, respectively. Additionally, assume that $p \leq \frac{k-2}{2}$ reachable pairs of terminal vertices have been formed already (including those pairs formed in step i). Then, it is possible to obtain a potential increase of at least $\frac{k-2p}{2d}$ in time step i .*

Proof. Observe that, since there are already p pairs formed amongst the terminals $u \in T$, there are exactly $k - 2p$ terminals that remain in T , i.e. $|T| = k - 2p$ (since any two terminals forming a pair become leftover vertices once the pair is formed). As $p \leq \frac{k-2}{2}$ implies $k - 2p \geq 2$, we know that there are still at least 2 terminals. We wish to find at least $\frac{k-2p}{2d}$ disjoint paths, such that each path has terminals as its endpoints, and that every other vertex in the path is a leftover vertex (we will refer to such a path as a *terminal path*). Let G_i^d denote the form of \mathcal{G}^d during the i -th time step; we proceed by computing a spanning tree S_i of G_i^d , selecting an arbitrary leaf, r , as S_i 's root, and forming paths in a bottom-up fashion via a greedy procedure. More specifically, we consider each vertex, $x \in S_i$, in reverse level order, so that we first examine those vertices that are furthest away from r , followed by those vertices second furthest away from r , and so on, until every x has been processed. On examining a vertex x , we consider the subtree of S_i rooted at x : if it contains 2 or more terminals that do not already belong to a path, we arbitrarily select two and take the path joining them in S_i to be one of our terminal paths, discarding the remaining terminals in that subtree. We claim that in this way, we use exactly 2 terminals for each path we form, and discard at most $d - 2$ terminals whenever we form a single path. To see this, observe that each vertex in S_i has at most $d - 1$ children (since r is a leaf), and consider the situation in which the vertex we are currently processing, x , is a terminal of degree d . If each of the subtrees rooted at x 's children contains exactly one terminal, then we will only be able to form one path, which must have x and one of the other $d - 1$ terminals (each of which lies alone in one of the subtrees rooted at x 's children) as its endpoints – the remaining $d - 2$ terminals will be discarded. This follows from the fact that if more than one terminal lay in the subtree rooted at any of x 's children, then they must already belong to a path; otherwise they would have already been discarded, as per our procedure. Therefore, on forming any path in S_i , we “use up” at most d terminals (2 terminals for the path, and at most $d - 2$ terminals are discarded). Observe now that, on processing r , it may also happen that there is a single unmatched terminal, y , in the subtree rooted at r ; in this case, y will be discarded. As a result, it follows that we are able to find at least $\frac{k-2p-1}{d}$ disjoint terminal paths in S_i . Moreover, $\frac{k-2p-1}{d} \geq \frac{k-2p}{2d}$ for $d > 0$ and $p \leq \frac{k-2}{2}$, which can be easily checked. Since $k - 2p \geq 2$ precisely when $p \leq \frac{k-2}{2}$, it follows that we are able to form at least $\frac{k-2p}{2d}$ disjoint terminal paths in S_i , as required.

Now, given a set of terminal paths obtained by following the aforescribed method, we wish to show that, per each path, we can obtain a +1 increase in potential. We require a way of doing so that ensures that not only are we able to increase the potential by a large enough

amount in every step, but that we are able to ensure the drop in potential experienced in any particular time step is limited. Specifically, we require a procedure with the following properties:

1. We can obtain a potential increase of $+1$ within any given terminal path.
2. For a single execution of our procedure on a given terminal path, we want the spread of exactly one of the path's endpoints to increase by 1, and the spread of the terminal at the opposite end of the path to remain the same.

Property (1) ensures that we are able to guarantee a potential increase of $\frac{k-2p}{2d}$ in each time step. Property (2) limits the number of home sets that any terminal vertex $u \in T$ can be added to in any particular step to 1. To this end, we specify now our procedure, demonstrating that both properties (1) and (2) are satisfied by the actions performed within each individual case.

► **Procedure (OBTAIN-POTENTIAL).** Consider an arbitrary time step, t , from which we began tracking the value of $\phi(\mathcal{G}^d)$. The input to the procedure is a terminal path, $Q_i \subseteq S_i$ (we omit the i from this notation from here onwards), obtained by applying the aforescribed greedy procedure to \mathcal{G}^d in step $i \geq t$ (i.e., by applying the greedy procedure to the graph \mathcal{G}_i^d). Let u and u' be the endpoints of Q . We proceed as follows: arbitrarily select one of u or u' (for argument's sake we select u), and begin examining the set $H_{v_j}(t, i)$ (again, we will omit the arguments from this notation) for every $v_j \in Q$ in the order in which they appear in Q (from u to u'). Let the x -th and last leftover vertex that we examine where $H_{v_j} = \{u\}$ be known as v_x ; the vertex we examine next will be known as v_{x+1} . We note that since, by our earlier assumption, all pairs that will form in step i have already been formed, the existence of such a v_x is guaranteed in all but one case. If u is adjacent to the vertex $v_1 \in Q$ such that $H_{v_1} \supseteq \{u^*\}$ with $u^* \neq u$, then clearly a pair is formed; a contradiction, since the greedy procedure for forming terminal paths does not consider terminals that already belong to pairs. Similarly, if $v_x = v_{|Q|-1}$, then $H_{v_{|Q|-1}} = \{u\}$ and u and u' form a pair in that step; again, a contradiction. Thus, the exceptional case, in which there is no such v_x , occurs when $H_{v_1} = \emptyset$: here, we can instantly obtain our potential increase for Q by adding u to H_{v_1} , whilst still satisfying properties (1) and (2). With these exceptions dealt with, we now distinguish between two main cases:

- (i) **Case 1: $H_{v_x} \cap H_{v_{x+1}} = \emptyset$.** We distinguish between two subcases:
 - (i) **Case 1.1: $H_{v_{x+1}} \neq \emptyset$.** Select an arbitrary u^* in $H_{v_{x+1}}$ and add u^* to H_{v_x} , giving our $+1$ potential increase (satisfying property (1)). We note that this is possible since $u^* \in H_{v_{x+1}}$ implies that $v_{x+1} \in R_{u^*}$, in which case an agent could move from u^* to v_{x+1} , and from v_{x+1} to v_x . It is therefore valid to select u^* from the home set of a vertex adjacent to v_x and add it to v_x 's home set since, by definition, v_x would be added to R_{u^*} in that step regardless. In the event that $u^* \notin \{u, u'\}$, remove u^* from $H_{v_{x+1}}$, and add u to $H_{v_{x+1}}$; this ensures that the potential associated with v_{x+1} does not decrease, whilst additionally ensuring that the spread of u^* does not increase with respect to Q (satisfying property (2)).
 - (ii) **Case 1.2: $H_{v_{x+1}} = \emptyset$.** In this case, we add u to $H_{v_{x+1}}$, giving our $+1$ potential increase (property (1)), whilst ensuring that only u is added to the home set of exactly one leftover vertex in Q (property (2)).
- (ii) **Case 2: $H_{v_x} \cap H_{v_{x+1}} \neq \emptyset$.** Again, we distinguish between two subcases:
 - (i) **Case 2.1: $u' \in H_{v_{x+1}}$.** Since $u' \in H_{v_{x+1}}$, we can simply add u' to H_{v_x} and we are done – this gets us our required $+1$ potential increase (property (1)), whilst

also satisfying property (2), since the only terminal that is added to a home set within Q is one of Q 's endpoints, u' .

- (ii) **Case 2.2: $u' \notin H_{v_{x+1}}$.** Select an arbitrary $u^* \in H_{v_{x+1}}$ (with $u^* \neq u, u'$), add u^* to H_{v_x} , and remove u^* from $H_{v_{x+1}}$. Now, continue following Q in the same direction, repeating the above process some $y - 1$ times, until in the y -th iteration we examine a vertex v_{x+y+1} , such that $u \notin H_{v_{x+y+1}}$. More generally, in the j -th iteration, we check the set $H_{v_{x+j+1}}$, select from it an arbitrary $u^* \neq u$, add u^* to $H_{v_{x+j}}$, and remove u^* from $H_{v_{x+j+1}}$. By removing u^* from $H_{v_{x+j+1}}$, we ensure that the spread of u^* does not increase with respect to Q , satisfying property (2) for all iterations 1 through $y - 1$ (notice that in each of these iterations, our potential value stays the same, since whenever we add a terminal to one leftover vertex's home set, we remove that terminal from another home set). If it happens that $H_{v_{x+j+1}} = \{u\}$ (i.e., we cannot select $u^* \neq u$), then we do not add any u^* to $H_{v_{x+j}}$ in that iteration; clearly property (2) is still satisfied in this situation. Once we begin the y -th iteration, if it happens that the set $H_{v_{x+y+1}} = \emptyset$, we simply add u to $H_{v_{x+y+1}}$ and we are done – in this case, both properties are trivially satisfied. Otherwise, select an arbitrary u^* from $H_{v_{x+y+1}}$, add u^* to $H_{v_{x+y}}$ (increasing the potential value of v_{x+y}), and remove u^* from $H_{v_{x+y+1}}$. Finally, add u to $H_{v_{x+y+1}}$; this uses the fact that $H_{v_{x+y}}$ is guaranteed to contain terminal u , and so we replace u^* with u , ensuring that the potential of vertex v_{x+y+1} does not decrease, whilst still ensuring that property (2) is satisfied, since only vertex u 's spread has increased by exactly 1. It is this final step that ensures property (1) is satisfied, since we increased the potential value of v_{x+y} in this iteration, but did not decrease the potential value of v_{x+y+1} .

The lemma follows by applying the greedy procedure (for forming $\frac{k-2p}{2d}$ disjoint terminal paths) in G_t^d , and supplying each path as input to the OBTAIN-POTENTIAL procedure. ◀

► **Lemma 14.** *Let \mathcal{G}^d be a degree-bounded temporal graph with a vertex set partitioned into parts T and L . Let t be the time at which tracking of the potential function began (so that $\phi^t(\mathcal{G}^d) = |L|$), and let i be the current time step. Then, the forming of a single reachable pair of terminal vertices, (u, u') , can cause the potential value to drop by at most $2l$, where $l = i - t$ is the number of steps that passed since we began tracking the value of $\phi(\mathcal{G}^d)$.*

Proof. Consider the specification of our OBTAIN-POTENTIAL procedure and note that, during any time step i (with $t \leq i \leq \tau$), each terminal vertex can belong to at most one disjoint terminal path, and that exactly one of the vertices in $\{u, u'\}$ (that form the endpoints of each terminal path) has its spread increase by 1. It is clear that when a reachable pair of terminal vertices (u, u') is formed, the worst case scenario is the following: both u and u' were added to the home set of a single leftover vertex in each of the l steps that have passed since time t . The lemma follows by observing that this scenario will result in an overall drop in potential of $2l$, since, by Definition 9, u and u' will be removed from the home set of all those leftover vertices $v \in L$ for which they belong to the set $H_v(t, i)$. ◀

► **Lemma 15.** *Let \mathcal{G}^d be a degree-bounded temporal graph with an underlying graph of order n , and let $V(\mathcal{G}^d)$ be partitioned into parts T and L , with $|T| = k$ and $|L| = n - k$. Then in $10 \cdot \frac{dn}{k}$ steps, at least $\frac{k}{20d}$ disjoint reachable pairs of terminal vertices are formed.*

Proof. Consider \mathcal{G}^d from time t onwards and assume that the opposite of our claim is true, so that less than $\frac{k}{20d}$ reachable pairs are formed in the space of $10 \cdot \frac{dn}{k}$ steps (i.e. less than $\frac{k}{20d}$ reachable pairs are formed within the time range t to $t + 10 \cdot \frac{dn}{k}$). Then, by the end of the

$(t + 10 \cdot \frac{dn}{k})$ -th step, there are at least $k - 2 \cdot \frac{k}{20d} = k - \frac{k}{10d} = \frac{10dk - k}{10d} = k \cdot \frac{(10d-1)}{10d} \geq 0.9k$ (for all $d \geq 1$) terminal vertices that have not yet formed a pair with another terminal vertex. It follows then, by Lemma 13, that the value of $\phi(\mathcal{G}^d)$ increases by at least $\frac{0.9k}{2d} \cdot \frac{10dn}{k} = \frac{9kdn}{2dk} = \frac{9n}{2} = 4.5n$ over those $10 \cdot \frac{dn}{k}$ steps. By Lemma 14, over the same period, the potential can decrease by at most $\frac{k}{20d} \cdot 2 \cdot \frac{10dn}{k} = \frac{k}{20d} \cdot \frac{20dn}{k} = \frac{20dkn}{20dk} = n$, since, by our earlier assumption, fewer than $\frac{k}{20d}$ pairs are formed. From this, it follows that the potential at the end of those $10 \cdot \frac{dn}{k}$ steps is at least $4.5n - n = 3.5n$. But this is a contradiction since, by Observation 12, $\phi^i(\mathcal{G}^d) \leq 3n$ for any $t \leq i \leq \tau$; the lemma follows. \blacktriangleleft

► Lemma 16. *Consider a degree-bounded temporal graph, \mathcal{G}^d , in an arbitrary time step, t , and let the vertices in $V(\mathcal{G}^d)$ be divided into parts T and L , with $|T| = k$ and $|L| = n - k$ at the beginning of time t . Then, there exists at least one vertex $v \in T$ such that an agent positioned at v at the start of time t can explore $\Theta(\log_d k)$ terminal vertices in $O(dn)$ steps.*

Proof. By application of Lemma 15 to part T , we are able to form at least $\frac{k}{20d}$ disjoint reachable pairs of terminal vertices in the space of $10 \cdot \frac{dn}{k}$ time steps. As a result, we obtain a set of $\frac{k}{20d}$ temporal walks, each of which has exactly one such reachable pair constituting its endpoints, with no walk taking any longer than $10 \cdot \frac{dn}{k}$ steps. At the end of these $\frac{k}{20d}$ walks are exactly $\frac{k}{20d}$ unique (since the pairs are disjoint) terminal vertices. We proceed by taking this smaller set of terminals as our new T and reapplying Lemma 15 at time $t + 10 \cdot \frac{dn}{k}$; as a result, we obtain $\frac{k/20d}{20d} = k/(20d)^2$ new walks, each of which has endpoints that form a reachable pair, and each of which takes no longer than $10 \cdot \frac{dn}{k/20d}$ steps. Notice now that since, via our second application of Lemma 15, we formed pairs amongst only those terminals that were reachable via one of the walks obtained from our initial application of Lemma 15, there are now at least $k/(20d)^2$ walks of length at most $\frac{10dn}{k} + \frac{10dn}{k/(20d)}$, each of which visits three terminal vertices; in other words, we are able to concatenate some walk obtained by our first application of Lemma 15, with some walk obtained by our second application, to construct a walk that visits three terminals. In this fashion, we claim that we are able to construct a walk that visits $\Theta(\log_d k)$ terminals.

Generally speaking, the i -th application of Lemma 15 produces at least $k/(20d)^i$ disjoint pairs of reachable unvisited vertices, in each of which one vertex can reach the other via a walk of length at most $10 \cdot (dn)/(k/(20d)^{i-1})$. Since each time we reapply Lemma 15, it is applied only to those terminals that are reachable via a walk computed by the previous application, it follows that there must be a sequence of i walks, one resulting from each application of Lemma 15, that can be concatenated (in the order in which they were produced) to form a walk that visits exactly $i + 1$ terminals (two terminals are visited by the first computed walk, then an additional terminal is explored by the walk obtained by each successive application). Observe that we are able to repeatedly apply Lemma 15 to T exactly $\lfloor \log_{20d} k \rfloor$ times until no more pairs can be formed in T . By taking the i -th application of Lemma 15 to be the $\lfloor \log_{20d} k \rfloor$ -th such application, it is clear that an agent following a walk constructed via the discussed concatenation method can explore $\lfloor \log_{20d} k \rfloor + 1 = \Theta(\log_d k)$ terminals. All that remains to be shown is that the length of any temporal walk, W , constructed in this way is of duration at most $O(dn)$. By our earlier discussion, a reachable pair of unvisited vertices, (u, u') , found as a result of the i -th application of Lemma 15, are separated by a walk with duration no longer than $10 \cdot dn/(k/(20d)^{i-1})$. Given that we apply Lemma 15 $\lfloor \log_{20d} k \rfloor$ times in order to obtain each of the shorter walks that are then concatenated to obtain W , we derive the following summation to bound the overall duration of W from above:

$$\sum_{i=0}^{\lfloor \log_{20d} k \rfloor} \left(10 \cdot \frac{dn}{k/(20d)^i} \right) = \frac{10dn}{k} \cdot \sum_{i=0}^{\lfloor \log_{20d} k \rfloor} (20d)^i \leq \frac{10dn}{k} \cdot 2k = O(dn),$$

as required. \blacktriangleleft

► **Theorem 17.** *Let \mathcal{G}^d be a degree-bounded temporal graph of order $|V(\mathcal{G}^d)| = n$. Then there exists an exploration schedule, W_{exp} , starting from a given vertex, s , such that W_{exp} 's arrival time, $\alpha(W_{exp})$, is $O(d \log d \cdot \frac{n^2}{\log n})$.*

Proof. Let the current time step be $t = 1$, and let $V(\mathcal{G}^d)$ be partitioned into sets T and L , with $|T| = k$ and $|L| = n - k$ at time t (for $k = n - 1$, since s is already explored). Initially, we wish to explore $\Theta(\log_d k)$ vertices during the first $O(dn)$ steps. To achieve this, we begin by constructing a temporal walk, W_1 , with departure time $\delta(W_1) = t = 1$. Let W_1 be the product of concatenating two walks, X_1 and Y_1 , in that order. We note that it is not guaranteed that s will be at the start of any walk computed via our first application of Lemma 16: therefore, we set X_1 to be of duration n , and apply Lemma 16 to \mathcal{G}^d at time $\delta(W_1) + |X_1| = \delta(W_1) + n$, initially setting $T = V(\mathcal{G}^d) - \{s\}$, and $L = \{s\}$, so that $T \cup L = V(\mathcal{G}^d)$. The walk resulting from that application of Lemma 16 will be known as Y_1 . In this way, we ensure, by Lemma 3 (Lemma 1 in [5]), that there is enough time for an agent to move, via X_1 , from s to whichever vertex Y_1 departs from. Since $|X_1| = n$, $|Y_1| = O(dn)$ (by Lemma 16), and, X_1 and Y_1 are compatible under walk concatenation, we are able to obtain the walk, W_1 , of duration $|W_1| = n + O(dn) = O(dn)$, that explores $\Theta(\log_d k)$ unique vertices.

We wish to apply the above process repeatedly, until only $k \leq \frac{n}{\log_{20d} n}$ vertices remain to be explored in \mathcal{G}^d . This can be achieved by constructing temporal walks W_i , such that each W_i is the concatenation of two walks X_i and Y_i . Let $\alpha(W_{i-1})$ be the arrival time of walk W_{i-1} . Each X_i is a walk with duration n and departure time $\delta(X_i) = \alpha(W_{i-1})$, that departs from the last vertex of W_{i-1} , and arrives at the first vertex of Y_i . We then define Y_i to be the temporal walk, exploring $\Theta(\log_d k)$ vertices, obtained via the i -th application of Lemma 16. We perform this i -th such application at time $\delta(W_{i-1}) + |X_i| = \delta(W_{i-1}) + n$, setting

$$T = V(\mathcal{G}^d) - (\{s\} \cup V(Y_1) \cup V(Y_2) \cup \dots \cup V(Y_{i-1})),$$

$L = V(\mathcal{G}^d) - T$ and $k = |T|$. Taking W_i to be the result of concatenating X_i and Y_i (in that order), it follows that each walk W_i departs from the last vertex of W_{i-1} , and so we can continually concatenate each W_i in the order that they are produced. The concatenation of each W_i explores an additional $\Theta(\log_d k)$ vertices; we continue this process until the number of unexplored vertices is less than or equal to $\frac{n}{\log_{20d} n}$.

In order to show now that our overall approach to exploration always produces schedules with arrival time $O(d \log d \cdot \frac{n^2}{\log n})$, we first show that $\log_{20d} k = \Theta(\log_d n)$ whenever we apply Lemma 16. As stated previously, we repeatedly apply the process discussed above until $k \leq \frac{n}{\log_{20d} n}$. This implies that whenever we apply Lemma 16, $k > \frac{n}{\log_{20d} n}$, giving that

$$\log_{20d} k > \log_{20d} \left(\frac{n}{\log_{20d} n} \right) = \log_{20d} n - \log_{20d} \log_{20d} n = \Theta(\log_d n).$$

From this, we can conclude that the duration of any W_i produced in the aforescribed manner visits $\Theta(\log_d n)$ vertices. Since we initially explore at least $n - \frac{n}{\log_{20d} n} = O(n)$ vertices using this method, it follows that we require the concatenation of $O(n) / \Theta(\log_d n)$ W_i 's, each of which is of duration $|W_i| = O(dn)$. Let the walk, resulting from the repeated concatenation of our W_i , be W_{exp}^1 ; it follows that W_{exp}^1 explores at least $n - \frac{n}{\log_{20d} n}$ vertices in at most

$$O(dn) \cdot \frac{O(n)}{\Theta(\log_d n)} = O\left(d \cdot \frac{n^2}{\log_d n}\right)$$

time steps. To deal with the remaining $\frac{n}{\log_{20d} n}$ vertices: let W_x be the last walk we produce via an application of Lemma 16, and assume it has arrival time $\alpha(W_x)$; clearly, W_{exp}^1 has arrival

time $\alpha(W_{exp}^1) = \alpha(W_x) = O(d \cdot \frac{n^2}{\log_d n})$. We apply Lemma 3 (Lemma 1 in [5]), computing a walk, W_{exp}^2 , which starts at time $\alpha(W_{exp}^1)$, and spends a total time of $O(n)$ steps visiting each of the remaining vertices. This gives W_{exp}^2 a total duration of $O(n) \cdot \frac{n}{\log_{20d} n} = O(\frac{n^2}{\log_d n})$ steps. Now, since W_{exp}^1 has respective departure and arrival times $\delta(W_{exp}^1) = 1$ and $\alpha(W_{exp}^1)$, and W_{exp}^2 departs at time $\alpha(W_{exp}^1)$ from the vertex at which W_{exp}^1 ends, it follows that they are compatible under walk concatenation. The result of their concatenation is a walk, W_{exp} , such that W_{exp} 's arrival time is:

$$\alpha(W_{exp}) = |W_{exp}^1| + |W_{exp}^2| = O(d \cdot \frac{n^2}{\log_d n}) + O(\frac{n^2}{\log_d n}) = O\left(d \cdot \frac{n^2}{\log_d n}\right) = O\left(d \log d \cdot \frac{n^2}{\log n}\right),$$

and the theorem follows. \blacktriangleleft

From the above, it follows that whenever $d \log d = o(\log n)$, then a degree-bounded temporal graph \mathcal{G}^d admits exploration schedules with arrival time $o(n^2)$. Moreover, Theorem 17 implies the following corollary:

► **Corollary 18.** *Let \mathcal{G}^d be a temporal graph whose maximum degree in every step is at most $d \geq 2$, and d is constant. Additionally, let s be a specified start vertex. Then there exists an exploration schedule, W_{exp} , which starts at vertex s and explores all vertices in \mathcal{G}^d , such that the arrival time of W_{exp} is $O(\frac{n^2}{\log n})$.*

We remark that our proof is constructive and implies a polynomial-time algorithm computing an exploration schedule with arrival time $O(d \log d \cdot \frac{n^2}{\log n})$, and thus also an $O(d \log d \cdot \frac{n}{\log n})$ -approximation algorithm for the TEXP problem when restricted to temporal graphs of bounded-degree.

4 Conclusion

Linear arrival times for the exploration of any static, undirected graph can be easily achieved by means of a depth-first search. The additional layer of complexity in the structure of temporal graphs, brought about by the potential for time-variance in the edge set, means that their exploration is not such a simple task: exploration schedules of general temporal graphs can require $\Theta(n^2)$ time steps.

Complementing previous results, which suggest that subjecting our input temporal graph to certain structural restrictions can improve arrival times, we have shown that by requiring the maximum degree in each time step to be bounded by d , one can guarantee arrival times of $O(d \log d \cdot \frac{n^2}{\log n})$. These results directly suggest a number of further questions: we would like to close the gap between the upper and lower bounds for exploring general degree-bounded temporal graphs – we suspect that there exists an upper bound lower than the one presented here, but it may also be the case that the current lower bound of $\Omega(n \log n)$ can be improved upon. The study of further restricted temporal graph classes, in order to establish bounds on the amount of time needed to explore them, remains an interesting question; for example, one might consider the class of temporal graphs in which only a constant number of edges are allowed to differ from each time step to the next.

A further question might ask precisely which structural properties a graph must possess in order for it to admit exploration schedules with arrival time $o(n^2)$? – it would be interesting to classify the graphs that can be explored in time strictly less than quadratic under these terms. Establishing how the computational complexity of TEXP changes whilst the problem is restricted to particular classes of graphs also presents an interesting direction; for which classes does the problem remain *NP*-hard to solve optimally, and how well can we approximate solutions for these restricted cases?

A number of other related questions also remain open. An example of one we find particularly interesting is as follows: how much quicker can general temporal graphs be explored if we allow an agent exploring the graph to make a move across two edges per time step, rather than one? We remark that the construction requiring $\Theta(n^2)$ steps to explore (given in [5]) requires only $O(n)$ steps in this two-move model. Establishing bounds for this model could provide further insight into the exploration of temporal graphs under the model considered throughout this paper.

References

- 1 Kenneth A. Berman. Vulnerability of scheduled networks and a generalization of Menger's theorem. *Networks*, 28(3):125–134, 1996. doi:10.1002/(SICI)1097-0037(199610)28:3<125::AID-NET1>3.0.CO;2-P.
- 2 Björn Brodén, Mikael Hammar, and Bengt J. Nilsson. Online and offline algorithms for the time-dependent TSP with time zones. *Algorithmica*, 39(4):299–319, 2004. doi:10.1007/s00453-004-1088-z.
- 3 Binh-Minh Bui-Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *Int. J. Found. Comput. Sci.*, 14(2):267–285, 2003. doi:10.1142/S0129054103001728.
- 4 A. Casteigts, P. Flocchini, Quattrociochi W., and N. Santoro. Time-varying graphs and dynamic networks. *IJPEDS*, 27(5):387–408, 2012.
- 5 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. In *42nd International Colloquium on Automata, Languages, and Programming (ICALP 2015), Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 444–455. Springer, 2015. doi:10.1007/978-3-662-47672-7_36.
- 6 Till Fluschnik, Hendrik Molter, Rolf Niedermeier, and Philipp Zschoche. Temporal graph classes: A view through temporal separators. *CoRR*, abs/1803.00882, 2018. arXiv:1803.00882.
- 7 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.
- 8 George B. Mertzios, Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis. Temporal network optimization subject to connectivity constraints. In *40th International Colloquium on Automata, Languages, and Programming (ICALP 2013), Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 657–668. Springer, 2013. doi:10.1007/978-3-642-39212-2_57.
- 9 Othon Michail. An introduction to temporal graphs: An algorithmic perspective. *Internet Mathematics*, 12(4):239–280, 2016. doi:10.1080/15427951.2016.1177801.
- 10 Othon Michail and Paul G. Spirakis. Traveling salesman problems in temporal graphs. *Theor. Comput. Sci.*, 634:1–23, 2016. doi:10.1016/j.tcs.2016.04.006.