The temporal explorer who returns to the base*

Eleni C. Akrida[a], George B. Mertzios[a], Paul G. Spirakis[b,c], Christoforos Raptopoulos[c]

[a]*Department of Computer Science, Durham University, UK*
[b]*Department of Computer Science, University of Liverpool, UK*
[c]*Department of Comp. Engineering & Informatics, University of Patras, Greece*

Abstract

We study here the problem of exploring a temporal graph when the underlying graph is a star. The aim of the exploration problem in a temporal star is finding a temporal walk which starts and finishes at the center of the star, and visits all leaves. We present a systematic study of the computational complexity of this problem, depending on the number k of time points where each edge can be present in the graph. We distinguish between the decision version StarExp(k), asking whether a complete exploration exists, and the maximization version MaxStarExp(k), asking for an exploration of the greatest possible number of edges. We fully characterize MaxStarExp(k) in terms of complexity. We also partially characterize StarExp(k), showing that it is in P for $k < 4$, but is NP-complete, for every $k > 5$. Finally, we partially characterize classes of "random" temporal stars which are, asymptotically almost surely, yes-instances and no-instances for StarExp(k).

Keywords: temporal networks, exploration, random input, edge availability

1. Introduction and motivation

A temporal graph is, roughly speaking, a graph that changes over time. Several networks, both modern and traditional, including social networks,

*A preliminary version appeared at the 11th International Conference on Algorithms and Complexity (CIAC 2019).

Email addresses: eleni.akrida@durham.ac.uk (Eleni C. Akrida), george.mertzios@durham.ac.uk (George B. Mertzios), p.spirakis@liverpool.ac.uk (Paul G. Spirakis), raptopox@ceid.upatras.gr (Christoforos Raptopoulos)

transportation networks, information and communication networks, can be modeled as temporal graphs. The common characteristic in all the above examples is that the network structure, i.e. the underlying graph topology, is subject to discrete changes over time. Temporal graphs naturally model such time-varying networks using time-labels on the edges of a graph to indicate moments of existence of those edges, while the vertex set remains unchanged. This formalism originates in the foundational work of Kempe et al. [32].

In this work, we focus in particular on temporal graphs where the underlying graph is a star graph and we consider the problem of exploring such a temporal graph starting and finishing at the center of the star. The motivation behind this is inspired from the well known Traveling Salesperson Problem (TSP). The latter asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin one?". In other words, given an undirected graph with edge weights where vertices represent cities and edges represent the corresponding distances, find a minimum-cost Hamiltonian cycle. However, what happens when the traveling salesperson has particular temporal constraints that need to be satisfied, e.g. (s)he can only go from city A to city B on Mondays or Tuesdays, or (s)he can only travel by train and, hence, needs to schedule his/her visit based on the train timetables? In particular, consider a traveling salesperson who, starting from his/her home town, has to visit $n-1$ other towns via train, always *returning to his/her own home town* after visiting each city. There are trains between each town and the home town only on specific times/days, possibly different for different towns, and the salesperson knows those times in advance. Can the salesperson decide whether (s)he can visit all towns and return to his/her own home town by a certain day?

1.1. Previous work.

Recent years have seen a growing interest in dynamic network studies. Due to its vast applicability in many areas, the notion of temporal graphs has been studied from different perspectives under various names such as time-varying [1, 25, 45], evolving [12, 19, 24], dynamic [16, 22, 28, 48, 49], and graphs over time [36]; for a recent attempt to integrate existing models, concepts, and results from the distributed computing perspective see the survey papers [13–15] and the references therein. Temporal data analytics, temporal flows, as well as various temporal analogues of known static graph concepts such as cliques, vertex covers, diameter, distance, connectivity and centrality have also been studied [2–6, 10, 30, 34, 35, 37, 46, 47].

Notably, temporal graph exploration has been studied before [11, 23, 31, 38]; Erlebach et al. [23] define the problem of computing a foremost exploration of all vertices in a temporal graph (TEXP), without the requirement of returning to the starting vertex. They show that it is NP-hard to approximate TEXP with ratio $O(n^{1-\varepsilon})$ for any $\varepsilon > 0$, and give explicit construction of graphs that need $\Theta(n^2)$ steps for TEXP. They also consider special classes of underlying graphs, such as the grid, as well as the case of random temporal graphs where edges appear in every step with independent probabilities. Michail and Spirakis [38] study a temporal analogue of TSP(1,2), where the objective is to explore the vertices of a complete directed temporal graph with edge weights from $\{1, 2\}$ with the minimum total cost; they derive several polynomial-time approximation algorithms, including a $(1.7+\varepsilon)$-approximation for the generic case of "temporal TSP(1,2)". Ilcinkas et al. [31] propose a $2^{O(\sqrt{\log n})}n$-time algorithm for exploring constantly connected dynamic graphs on an underlying cactus graph, also showing a lower bound of $2^{\Omega(\sqrt{\log n})}n$ for the algorithm. Bodlaender and van der Zanden [11] show that exploring temporal graphs of small pathwidth is NP-complete; they start from the problem that we define in this paper[1], which we prove is NP-complete, and give a reduction to the problem of exploring temporal graphs of small pathwidth.

We focus here on the exploration of temporal stars, inspired by the TRAVELING SALESPERSON paradigm where the salesperson returns to his/her base after visiting every city. The TRAVELING SALESPERSON PROBLEM is one of the most well-known combinatorial optimization problems, which still poses great challenges despite having been intensively studied for more than sixty years. For the SYMMETRIC TSP, where the given graph is undirected (as is the case for the temporal version of the problem that we consider here) and the edge costs obey the triangle inequality, the best known approximation algorithm is still the celebrated 3/2 – approximation of Christofides [17], despite forty years of intensive efforts to improve it. Only recently, Gharan et al. [27] proved that the special case of GRAPHIC TSP, where the costs correspond to shortest path distances of some given graph, can be approximated

[1] A preliminary version of this paper appeared publicly in ArXiv on 12^{th} May 2018 (`https://arxiv.org/pdf/1805.04713.pdf`).

within $3/2 - \varepsilon$, for a small constant $\varepsilon > 0$ (which was further improved by subsequent works, e.g. [42]). For the ASYMMETRIC TSP where paths may not exist in both directions or the distances might be different depending on the direction, the $O(\log n)$-approximation of [26] was the best known for almost three decades, improved only recently to $O(\log n/\log\log n)$ [7] and then even more recently with the breakthrough result of an $O(1)$-approximation [44]. Online TSP-related problems as well as versions of TSP where each node must be visited within a given time window or following a particular timetable have been recently studied [9, 29, 41]. Various other TSP variations have also been studied over the years, including polynomially-solvable cases [20, 21].

1.2. The model and definitions.

It is generally accepted to describe a network topology using a graph, the vertices and edges of which represent the communicating entities and the communication opportunities between them, respectively. Unless otherwise stated, we denote by n and m the number of vertices and edges of the graph, respectively.

We consider graphs whose edge availabilities are described by sets of positive integers (labels), one set per edge.

Definition 1 (Temporal Graph)**.** *Let $G = (V, E)$ be a graph. A temporal graph on G is a pair (G, L), where $L : E \to 2^{\mathbb{N}}$ is a time-labeling function, called a* labeling *of G, which assigns to every edge of G a set of (discrete) time labels. The labels of an edge are the* (discrete) time instances ("days") *at which it is available.*

We call G the *underlying graph* and we call the largest label assigned by the function L to any edge of G the *lifetime* of the temporal graph.

For the majority of the following sections, we focus on the above standard model of temporal graphs (also studied in, e.g. [4, 5]), where the labels of an edge may be positive integers. However, in the final section we will also discuss a similar model of temporal graphs where the labels are real numbers selected from the continuous time interval $[0, 1]$. The main reason for studying this model is that it simulates the behavior of the system when the lifetime of the temporal graph tends to infinity and, thus, there are virtually no repetitions of labels on an edge of the underlying graph.

In this paper, we focus on temporal graphs whose underlying graph is an undirected star, i.e. a connected graph of $m = n - 1$ edges which has $n - 1$ leaves, i.e. vertices of degree 1.

Definition 2 (Temporal Star)**.** *A temporal star is a temporal graph (G_s, L) on a star graph $G_s = (V, E)$. Henceforth, we denote by c the center of G_s, i.e. the vertex of degree $n - 1$.*

Definition 3 (Time edge)**.** *Let $e = \{u, v\}$ be an edge of the underlying graph of a temporal graph and consider a label $l \in L(e)$. The pair $(\{u, v\}, l)$ is called* time edge.

A basic assumption that we follow here is that when a message or an entity passes through an available link at time (day) t, then it can pass through a subsequent link only at some time (day) $t' > t$ and only at a time at which that link is available. The problem of exploring a star graph becomes trivial otherwise, i.e. in the "non-strict" setting where labels on consecutive edges of the journey only have to be non-decreasing

Definition 4 (Journey)**.** *A* temporal path *or* journey *j from a vertex u to a vertex v* (*(u, v)*-journey) *is a sequence of time edges $(\{u, u_1\}, l_1)$, $(\{u_1, u_2\}, l_2)$, $\ldots$, $(\{u_{k-1}, v\}, l_k)$, such that $l_i < l_{i+1}$, for each $1 \leq i \leq k - 1$. We call the last time label, l_k,* arrival time *of the journey.*

Given a temporal star (G_s, L), on the one hand we investigate the complexity of deciding whether G_s is *explorable*: we say that (G_s, L) is explorable if there is a journey starting and ending at the center of G_s that visits every node of G_s. Equivalently, we say that there is an *exploration* that *visits* every node, and *explores* every edge, of G_s. On the other hand, we investigate the complexity of computing an exploration schedule that visits the greatest number of edges. A (partial) exploration of a temporal star is a journey J that starts and ends at the center of G_s which visits some nodes of G_s; its size $|J|$ is the number of nodes of G_s that are visited by J, where each vertex is only accounted for once even if it is visited multiple times. We, therefore, identify the following problems:

STAREXP(k)

Input: A temporal star (G_s, L) such that every edge has at most k labels.
Question: Is (G_s, L) explorable?

MaxStarExp(k)

Input: A temporal star (G_s, L) such that every edge has at most k labels.
Output: A (partial) exploration of (G_s, L) of *maximum* size.

Note that the case where one edge e of the input temporal star has only one label is degenerate. Indeed, in the decision variant (i.e. StarExp(k)) we can immediately conclude that (G_s, L) is a no-instance as this edge cannot be explored; similarly, in the maximization version (i.e. MaxStarExp(k)) we can just ignore edge e for the same reason. We say that we "enter" an edge $e = \{c, v\}$ of (G_s, L) when we cross the edge from c to v at a time on which the edge is available. We say that we "exit" e when we cross it from v to c at a time on which the edge is available. Without loss of generality we can assume that, in an exploration of (G_s, L), the entry to any edge e is followed by the exit from e at the earliest possible time (after the entry). That is, if the labels of an edge e are $l_1, l_2, \dots, l_k$ and we enter e at time l_i, we exit at time l_{i+1}. The reason is that, waiting at a leaf (instead of exiting as soon as possible) does not help in exploring more edges; we are better off returning to the center c as soon as possible.

In order to solve the problem of exploring as many edges of a temporal star as possible, we define here the Job Interval Selection Problem where each job has at most k associated intervals (JISP(k)), $k \geq 1$.

Job Interval Selection Problem - JISP(k) [43]

Input: n jobs, each described as a set of at most k intervals on the real line.
Output: A schedule that executes as many jobs as possible; to execute a job one needs to select one interval associated with the job. To execute several jobs, the intervals selected for the jobs must not overlap.

Notice that every edge e with labels $l_1, l_2, \dots, l_k$ can be seen as a job to be scheduled where the corresponding intervals are $[l_1, l_2], [l_2, l_3], \dots, [l_{k-1}, l_k]$, hence MaxStarExp($k$) is a special case of JISP($k-1$). JISP(k) is a well-studied problem in the Scheduling community with several known complexity results, some of which carry over to the MaxStarExp(k) problem. In particular, Spieksma [43] showed a (1/2)-approximation for the problem, later improved to a (1/1.582)-approximation by Chuzhoy et al. [18]. This immediately implies a (1/1.582)-approximation algorithm for MaxStarExp(k); we use the latter to conclude on the APX-completeness[2] of MaxStarExp(k) after proving that the latter is APX-hard for any $k \geq 4$. Note here that JISP(k) was shown [43] to be APX-hard for any $k \geq 2$, but since MaxStarExp(k) is a special case of JISP($k-1$), its hardness does not follow from the already known results. In fact, we show that MaxStarExp(3) -which is a special case of JISP(2)- is polynomially solvable.

1.3. Our contribution.

In this paper we do a systematic study of the computational complexity landscape of the temporal star exploration problems StarExp(k) and MaxStarExp(k), depending on the maximum number k of labels allowed per edge. As a warm-up, we first prove in Section 2 that the maximization problem MaxStarExp(3), i.e. when every edge has at most three labels per edge, can be efficiently solved in $O(n \log n)$ time; sorting the labels of the edges is the dominant part in the running time.

In Section 3 we prove that, for every $k \geq 6$, the decision problem StarExp(k) is NP-complete and, for every $k \geq 4$, the maximization problem MaxStarExp(k) is APX-hard, and thus it does not admit a Polynomial-Time Approximation Scheme (PTAS), unless P = NP. These results are proved by reductions from special cases of the satisfiability problem. The APX-hardness result is complemented by a (1/1.582)-approximation algorithm for MaxStarExp(k) for any k, which implies that MaxStarExp(k) is APX-complete for $k \geq 4$. This approximation algorithm carries over from an approximation for the Job Interval Selection Problem [18].

Finally, in Section 4 we study the problem of exploring a temporal star whose edges receive k random labels each. Here, we distinguish between two similar models for which we show lower and upper bounds on the number of labels per edge needed for full exploration of the resulting temporal star graph with high probability. The first model assigns k labels to each edge of G_s, independently of other edges, and each label is chosen uniformly at random and independently from the set of positive integers $\{1, 2, \ldots, \alpha\}$, for some $\alpha \in \mathbb{N}$. For this first model, we partially characterize the classes of temporal stars which, asymptotically almost surely, admit a complete (resp. admit no complete) exploration. In the second model, each label of each edge is a real

[2] APX is the complexity class of optimization problems that allow constant-factor approximation algorithms.

number[3] chosen uniformly at random and independently from the continuous interval $[0, 1]$. For the second model, we provide a lower bound for the number of labels per edge needed for full exploration of the star with high probability. We also discuss, in Section 4, the relation between the two models and the implications of a proven upper/lower bound on the number of labels per edge needed for explorability in one model on the other.

2. Efficient algorithm for $k \leq 3$ labels per edge

In this section we show that, when every edge has two or three labels, a maximum size exploration in (G_s, L) can be efficiently found in $O(n \log n)$ time. To do that, we reduce our problem to the Interval Scheduling Maximization Problem (ISMP).

Interval Scheduling Maximization Problem (ISMP)
Input: A set of intervals, each with a start and a finish time.
Output: A set of non-overlapping intervals of maximum size.

We can then apply a known greedy algorithm that finds an optimal solution for ISMP; the basic idea of this algorithm is to order the set of intervals in increasing order of finish time and then "greedily" process them in one pass, selecting as large a compatible subset as it can.

By proving that MaxStarExp(3) is solvable in $O(n \log n)$ time, we clearly also prove that the decision variation of the problem, i.e. StarExp(3), can also be solved within the same time bound. We give here the proof for $k = 3$ labels, which also covers the case of $k = 2$.

Theorem 1. MaxStarExp(3) *can be solved in* $O(n \log n)$ *time.*

Proof. We show that MaxStarExp(3) is reducible to ISMP. Given (G_s, L) we construct in linear time a set I of at most $2(n-1)$ intervals as follows: all edges of (G_s, L) with a single label can be ignored as they can not be explored in any exploration of (G_s, L); for every edge e of (G_s, L) with labels $l_e < l'_e$ we create a single *closed* time interval, $[l_e, l'_e]$; for every edge e of (G_s, L) with labels $l_e < l'_e < l''_e$ we create two *closed* time intervals, $[l_e, l'_e]$ and $[l'_e, l''_e]$.

We may now compute a maximum size subset I' of I of non-conflicting (i.e. disjoint) time intervals, using the greedy optimal algorithm for

[3] This is distinct from the rest of the paper, where we assume the standard model of integer labels.

ISMP [33]. It suffices to observe that no two intervals associated with the same edge will ever be selected in I', as any two such intervals are non-disjoint; indeed, two intervals associated with the same edge e are of the form $[l_e, l'_e]$ and $[l'_e, l''_e]$, hence they overlap at the single time point l'_e.

So a maximum-size set I' of non-overlapping intervals corresponds to a maximum-size exploration of (G_s, L) (in fact, of the same size as the size of I'). Also, we may indeed solve StarExp(3) by checking whether $|I'| = n-1$ or not. The above works in $O(n \log n)$ time [33]. □

3. Hardness for $k \geq 4$ labels per edge

In this section we show that, whenever $k \geq 6$, StarExp(k) is NP-complete. Furthermore, we show that MaxStarExp(k) is APX-hard for $k \geq 4$. Thus, in particular, MaxStarExp(k) does not admit a Polynomial-Time Approximation Scheme (PTAS), unless P = NP. In fact, due to a known polynomial-time constant-factor approximation algorithm for JISP(k) [18], it follows that MaxStarExp(k) is also APX-complete.

3.1. StarExp(k) *is NP-complete for $k \geq 6$ labels per edge*

We prove our NP-completeness result through a reduction from a special case of 3SAT, namely 3SAT(3), which is known to be NP-complete [39].

> 3SAT(3)
> **Input:** A boolean formula in CNF with variables $x_1, x_2, \ldots, x_p$ and clauses $c_1, c_2, \ldots, c_q$, such that each clause has at most 3 literals, and each variable appears in at most 3 clauses.
> **Question:** Is the formula satisfiable?

Intuition and overview of the reduction:. Given an instance F of 3SAT(3), we shall create an instance (G_s, L) of StarExp(k) such that F is satisfiable if and only if (G_s, L) is explorable. Henceforth, we denote by $|\tau(F)|$ the number of clauses of F that are satisfied by a truth assignment τ of F. Without loss of generality we make the following assumptions on F. Firstly, if a variable occurs only with positive (resp. only with negative) literals, then we trivially set it to *true* (resp. $false$) and remove the associated clauses. Furthermore, without loss of generality, if a variable x_i appears three times in F, we assume that it appears once as a negative literal $\neg x_i$ and two times as a positive literal x_i; otherwise we rename the negation with a new variable.

Similarly, if x_i appears two times in F, then it appears once as a negative literal $\neg x_i$ and once as a positive literal x_i.

Before we move on to the specifics of our reduction, we shall introduce the intuition behind it. (G_s, L) will have one edge corresponding to each clause of F, and three edges (one "primary" and two "auxiliary" edges) corresponding to each variable of F. We shall assign labels in pairs to those edges so that it is possible to explore an edge only by using labels from the same pair to enter and exit the edge; for example, if an edge e is assigned the pairs of labels l_1, l_2 and l_3, l_4, with $l_1 < l_2 < l_3 < l_4$, we shall ensure that one cannot enter e with, say, label l_2 and exit with, say, label l_3, by introducing the above-mentioned "auxiliary" edges and assigning appropriate labels to them to enforce the exploration of e using one of the desired pairs of labels.

In particular, for the "primary" edge corresponding to a variable x_i we will assign to it two pairs of labels, namely $(\alpha i - \beta, \alpha i - \beta + \gamma)$ and $(\alpha i + \beta, \alpha i + \beta + \gamma)$, for some $\alpha, \beta, \gamma \in \mathbb{N}$. The first (entry,exit) pair corresponds to setting x_i to false, while the second pair corresponds to setting x_i to true. We shall choose α, β, γ so that the entry and exit from the edge using the first pair is not *conflicting* with the entry and exit using the second pair, i.e. $(\alpha i - \beta, \alpha i - \beta + \gamma)$ and $(\alpha i + \beta, \alpha i + \beta + \gamma)$ do not overlap.

Then, to any edge corresponding to a clause c_j that contains x_i unnegated, we shall assign an (entry, exit) pair of labels $(\alpha i - \delta, \alpha i - \delta + \varepsilon)$, choosing $\delta, \varepsilon \in \mathbb{N}$ so that $(\alpha i - \delta, \alpha i - \delta + \varepsilon)$ is conflicting with the $(\alpha i - \beta, \alpha i - \beta + \gamma)$ pair of labels of the edge corresponding to x_i, which is associated with $x_i = false$ but not conflicting with the $(\alpha i + \beta, \alpha i + \beta + \gamma)$ pair. If x_i is false in F then c_j cannot be satisfied through x_i so we should not be able to explore a corresponding edge via a pair of labels associated with x_i. If c_j contains x_i negated, we shall assign to its corresponding edge an (entry, exit) pair of labels $(\alpha i + \zeta, \alpha i + \zeta + \theta)$, choosing $\zeta, \theta \in \mathbb{N}$ so that the latter is in conflict with the $(\alpha i + \beta, \alpha i + \beta + \gamma)$ pair of labels of the edge corresponding to x_i, which is associated with $x_i = true$ but not in conflict with the $(\alpha i - \beta, \alpha i - \beta + \gamma)$ pair. If x_i is true in F then c_j cannot be satisfied through $\neg x_i$ so we should not be able to explore a corresponding edge via a pair of labels associated with $\neg x_i$.

Finally, for every variable x_i we also introduce two additional "auxiliary" edges: the first one will be assigned the pair of labels $(\alpha i, \alpha i + \xi)$, $\xi \in \mathbb{N}$, so that it is not conflicting with any of the above pairs – the reason for introducing this first auxiliary edge is to avoid entering and exiting an edge corresponding to some variable x_i using labels from different pairs. The second auxiliary edge for variable x_i will be assigned the pair of labels $(\alpha i + \chi, \alpha i + \chi + \psi)$, $\chi, \psi \in \mathbb{N}$, so that it is not conflicting with any of the above pairs – the reason for introducing this edge is to avoid entering an edge that corresponds to some clause c_j using a label associated with some variable x_i and exiting using a label associated with a *different* variable $x_{i'}$.

The reduction:. For the reduction from 3SAT(3) to StarExp(k) we select constants $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \theta, \xi, \chi, \psi$ so that all the requirements mentioned above regarding the conflicts of different pairs of labels are satisfied. In particular, given an instance F of 3SAT(3), we create in polynomial time the following instance (G_s, L) of StarExp(k):

- For every variable x_i, $i = 1, 2, \ldots, p$, create an edge e_i (the "primary" edge for x_i) with labels $50i - 10$, $50i - 7$, $50i + 10$, and $50i + 13$. The pair of labels $50i - 10$, $50i - 7$ of e_i represent the assignment $x_i = false$ and the pair of labels $50i + 10$, $50i + 13$ represent the assignment $x_i = true$. More specifically, entry in e_i with label $50i - 10$ and exit from e_i with label $50i - 7$ represents $x_i = false$ (and similarly for the other pair of labels and the assignment $x_i = true$).

- For every variable x_i, $i = 1, 2, \ldots, p$, also create an edge e'_i (the first "auxiliary" edge for x_i) with labels $50i$ and $50i + 1$. The labels on e'_i ensure that we do not enter e_i with a label associated with the assignment $x_i = false$ and exit from e_i with a label associated with the assignment $x_i = true$; in an exploration of (G_s, L), this will not occur since we must explore both e'_i and e_i, and this only happens if we enter and exit e_i using labels associated with the same truth assignment for the variable x_i.

- For every variable x_i, $i = 1, 2, \ldots, p$, also create an edge e''_i (the second "auxiliary" edge for x_i) with labels $50i + 15$ and $50i + 16$. The labels on e''_i ensure that we do not enter and exit edges associated with clauses of F (see bullet point below) using pairs of labels that are associated with different variables.
  We refer the reader to Figure 1 for a sketch time-line indicating all the relevant times relating to variable x_i.

- For every clause c_j, $j = 1, 2, \ldots, q$, create an edge e_{p+j} with the following labels:

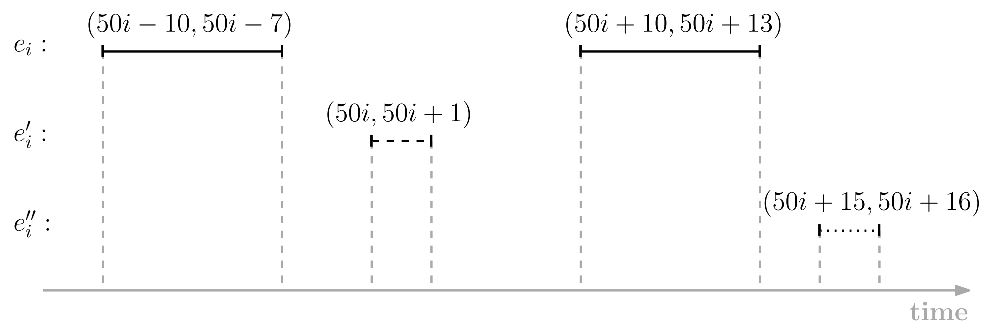


Figure 1: The time windows relating to variable x_i. There are two possible windows of exploration of the edge e_i, a single window of exploration for the edge e'_i (dashed line), and a single window of exploration for the edge e''_i (dotted line).

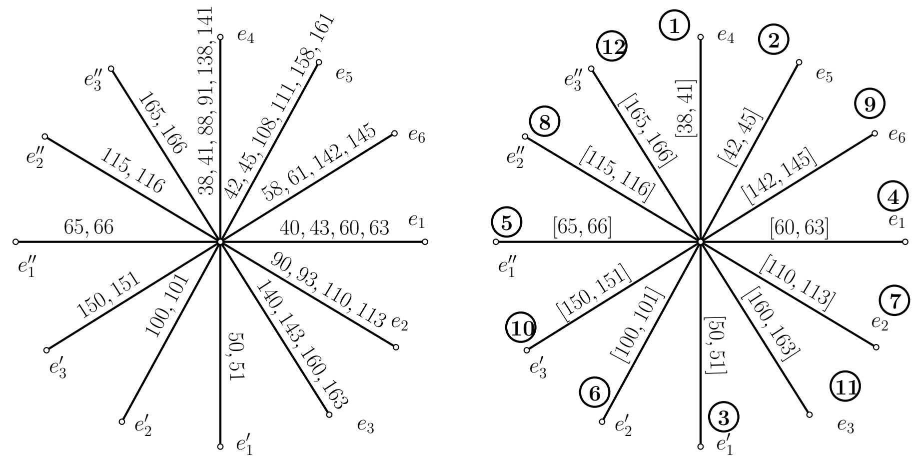
- For every variable x_i that appears unnegated for the *first*[4] time in C, add two labels $50i - 12$ and $50i - 9$.
- For every variable x_i that appears unnegated for the *second*[5] time in C, add two labels $50i - 8$ and $50i - 5$. Note here that both (entry,exit) pairs $(50i - 12, 50i - 9)$ and $(50i - 8, 50i - 5)$ are conflicting with the (entry,exit) pair $(50i - 10, 50i - 7)$ of the edge e_i that is associated with the assignment $x_i = false$.
- For every variable x_i that appears negated, add two labels $50i + 8$ and $50i + 11$. Note that the (entry,exit) pair $(50i + 8, 50i + 11)$ is conflicting with the (entry,exit) pair $(50i + 10, 50i + 13)$ of the edge e_i that is associated with the assignment $x_i = true$.

The reader is referred to Figure 2 for an example construction.

Notice that the (entry,exit) pairs on edges associated with different clauses are not conflicting if we pair them in ascending order, e.g. if an edge has labels l_1, l_2, l_3, l_4 we pair them into (l_1, l_2), (l_3, l_4); that is because each variable appears at most twice unnegated and once negated. The following lemmas are needed for the proof of NP-completeness (Theorem 2).

[4] We consider here the order $c_1, c_2, \ldots, c_q$ of the clauses of C; we say that x_i appears unnegated for the *first* time in some clause c_μ if $x_i \notin c_m,\ m < \mu$.

[5] Again, we consider the order $c_1, c_2, \ldots, c_q$ of the clauses of C; we say that x_i appears unnegated for the *second* time in some clause c_μ if $\exists m < \mu$, such that $x_i \in c_m$.



(a) (b)

Figure 2: The temporal star constructed for the formula $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3)$. Setting x_1 to true, x_2 to true and x_3 to true yields a satisfying truth assignment whose corresponding exploration is indicated in (b), where the numbers in the circles indicate the order over time of the exploration of each edge.

Lemma 1. *There exists a (partial) exploration J of (G_s, L) of maximum size which explores all the edges e_i, e'_i, e''_i, $i = 1, 2, \ldots, p$.*

Proof. Let J be a (partial) exploration of (G_s, L) of maximum size. Without loss of generality, we may assume that J explores every edge at most once.

We will show that one may also assume that any edge of the form e_{p+j} corresponding to the clause c_j, $j = 1, \ldots, q$, that is explored by J is explored via (entry,exit) pairs associated with the same literal of c_j. Indeed, let $c_j = (l_\pi \vee l_\rho \vee l_\sigma)$ be a clause whose corresponding edge is visited by J, where the literal l_π (resp. l_ρ and l_σ) is x_π (resp. x_ρ and x_σ) or $\neg x_\pi$ (resp. $\neg x_\rho$ and $\neg x_\sigma$), for some $\pi = 1, \ldots, p$ (resp. $\rho = 1, \ldots, p$ and $\sigma = 1, \ldots, p$).

Let $\alpha_1, \ldots, a_6$ be the labels of e_{p+j}. If J explores e_{p+j} using (α_2, α_3) (resp. (α_4, α_5)), then there exists an edge e''_r, for some $r = 1, \ldots, p$, which is not explored by J, by construction of the edges of the form e''_i. Then, one can create a (partial) exploration of the same size as J as follows: starting from J swap the exploration of e_{p+j} with the exploration of e''_r. Note that there is no other edge that is explored by J using a time window that overlaps with that of the exploration of e''_r, since it would also be conflicting with the window used by J to explore e_{p+j}. By iteratively swapping edges of the form e_{p+j}

- that are explored using (entry, exit) pairs associated with different literals - with edges of the form e''_r, we result in a (partial) exploration J', of the same size as J, such that any edge of the form e_{p+j} that is explored by J' is explored via an (entry,exit) pair of labels associated with the same literal of c_j. Note that no exploration window of any edge e'_i, e''_i overlaps with any exploration window of any edge e_{p+j} in J'. In fact, since J' is of maximum size, all edges of the form e''_i must be explored by J'.

We will now show that we may also assume that all edges of the form e'_i are explored by a maximum size (partial) exploration of (G_s, L). Assume that an edge e'_i is not explored by J', $i = 1, \ldots, p$. Then it must be that the edge e_i corresponding to the variable x_i is explored by J' using the pair $(50i-7, 50i+10)$ (as this is the only possible conflicting exploration window). Construct an exploration J'' as follows: starting from J',

1. replace $(50i - 7, 50i + 10)$ by $(50i + 10, 50i + 13)$ as the exploration window of e_i, and

2. add e'_i to the exploration using its window $(50i, 50i + 1)$.

Notice that step 1 is indeed possible without causing conflicts with other edges: let c_α be the clause containing $\neg x_i$ (so, the edge $e_{p+\alpha}$ has been assigned, amongst others, the labels $50i+8$, $50i+11$); then, since J' explores e_i using $(50i-7, 50i+10)$ it must be that $e_{p+\alpha}$ is not explored by J' -if explored at all- using $(50i+8, 50i+11)$. So, swapping the windows as shown in step 1 is possible without conflicts. Now, notice that J'' has size larger than the size of J' which is a contradiction. Therefore, e_i cannot be explored by J' (or any maximum size exploration of (G_s, L)) using $(50i - 7, 50i + 10)$, and e'_i must be explored by J', for all $i = 1, \ldots, p$.

It remains to show that all edges e_i, $i = 1, \ldots, p$, are explored by J'. Assume that there is an edge e_i, for some $i = 1, \ldots, p$, that is not explored by J' and let c_α be the clause that contains $\neg x_i$. The only way that e_i cannot be explored by J' is if J' explores edges that cause a conflict with both exploration windows $(50i-10, 50i-7)$ and $(50i+10, 50i+13)$ of e_i. In fact, if e_i is not explored by J' then it must be that $e_{p+\alpha}$ is explored using the exploration window $(50i + 8, 50i + 11)$. Then we can create J'' of same size as J', starting from J', by removing $e_{p+\alpha}$ from the exploration and adding e_i to the exploration, exploring it using the window $(50i + 10, 50i + 13)$. This way, one can create a maximum size exploration that contains all edges e_i, $i = 1, \ldots, p$.

We conclude that there can always be found an exploration of maximum size which explores all edges $e_i, e'_i, e''_i,\ i = 1, \ldots, p$, which completes the proof of the lemma. $\square$

Lemma 2. *There exists a truth assignment τ of F with $|\tau(F)| \geq \beta$ if and only if there exists a (partial) exploration J of (G_s, L) of size $|J| \geq 3p + \beta$.*

Proof. ($\Rightarrow$) Assume that there is a truth assignment τ that satisfies β clauses of F. We give a (partial) exploration J of (G_s, L), of size $3p + \beta$, as follows. First, we add to J all the edges $e'_i, e''_i,\ i = 1, 2, \ldots, p$; these are $2p$ edges in total and can only be explored one way as they have each been assigned two labels. Then, we add to J all edges $e_i,\ i = 1, 2, \ldots, p$ which are p edges in total; we explore each e_i depending on the value of x_i in τ, namely if $x_i = true$ we explore e_i using the pair $(50i + 10, 50i + 13)$, and if $x_i = false$ we explore e_i using the pair $(50i - 10, 50i - 7)$. Now, consider an arbitrary clause c_j of F that is satisfied in τ, i.e. it has at least one true literal which is of the form x_i or $\neg x_i$, for some $i = 1, 2, \ldots, p$. If $x_i = true$ then we explore e_{p+j} using the pair of labels that corresponds to the unnegated appearance of x_i in c_j – depending on whether x_i appears unnegated for the first or the second time in c_j, this pair is $(50i - 12, 50i - 9)$ or $(50i - 8, 50i - 5)$, respectively. If $\neg x_i = true$ then we explore e_{p+j} using the pair of labels $(50i + 8, 50i + 11)$ that corresponds to the negated appearance of x_i in c_j. As there are at least β satisfied clauses of F in τ, we have added at least β extra edges to J. It has already been established earlier in this section that all the (entry,exit) pairs chosen for the exploration of the edges that we added in J are pairwise non-conflicting. So, J is a (partial) exploration of (G_s, L) which explores at least $3p + \beta$ edges.

($\Leftarrow$) Assume that there is a (partial) exploration J of (G_s, L) which explores at least $3p+\beta$ edges. By Lemma 1, there is a (partial) exploration J of (G_s, L) of maximum size which explores all edges e_i, e'_i, e''_i, for $i = 1, 2, \ldots, p$. It holds that $|J| \geq 3p + \beta$, since J is of maximum size and we already know that there exists an exploration of size at least $3p + \beta$; also, J explores at least β edges that are associated with clauses of F, since there are in total $3p$ edges in (G_s, L) that are *not* associated with clauses of F. We can construct a truth assignment τ of F which satisfies at least β clauses as follows.

We check the (entry,exit) pairs of exploration in J of the edges that correspond to clauses of F. The entry and exit labels must be associated with the same variable x_i, otherwise there would be conflict with the exploration of the respective edge e''_i. So, for each such edge, we set the variable x_i

associated to the chosen (entry,exit) pair to true if the (entry,exit) pair is $(50i-12, 50i-9)$ or $(50i-8, 50i-5)$, i.e. if the pair corresponds to an unnegated appearance of x_i; we set x_i to false if the chosen (entry,exit) pair is $(50i+8, 50i+11)$. Since J explores at least β edges that are associated with clauses of F, we have now set the value of at least β variables of F to true or false. For any remaining variable x_i, the value of which has not already been set via the above process, we may set x_i to false (resp. true) if J explores the primary edge e_i associated with x_i using the pair of labels $50i-10$, $50i-7$ (resp. $50i+10$, $50i+13$). It is easy to see that the resulting truth assignment satisfies at least β clauses of F, each one associated with an edge e_{p+j}, $j \in \{1, 2, ..., q\}$, that is explored in J. □

We move on to the main theorem of the section.

Theorem 2. StarExp(k) *is NP-complete for every* $k \geq 6$.

Proof. It is easy to see that StarExp(k) is in NP, for every k. We may verify any solution, i.e. an exploration given as a list of edges and associated entry and exit times, in polynomial time by:

1. checking that the exploration visits all $O(n)$ vertices,
2. checking that the exploration enters and exits all $O(n)$ edges on existing edge-labels (we would need to check at most $k(n-1)$ labels in total), and
3. checking that the sorted list that contains all the entry and exit times together is such that, for every $i = 1, 2, \ldots, n-1$, the labels located at positions $2i-1$ and $2i$ in the ordered list are associated with the same edge; this last condition verifies that there are no overlaps in the given exploration windows of the solution.

An immediate corollary of Lemma 2 is that F is satisfiable if and only if (G_s, L) is explorable. Therefore, since the constructed instance (G_s, L) from the reduction has at most 6 labels per edge, it follows that StarExp(6) is NP-complete.

To extend this result to the NP-completeness of StarExp(k) also for values $k \geq 6$, it suffices to add to the constructed instance (G_s, L) an "artificial" edge e^* that only contains labels that are much larger than any of the labels on the other edges of (G_s, L). If F is satisfiable then the exploration of (G_s, L) is as described previously, with the addition of exploring e^* using any of its exploration windows. This is possible, since none of those will be conflicting with any window of any other edge. Conversely, if (G_s, L) is explorable, then the exploration of e^* can be ignored regarding the satisfiability of F since it overlaps with no other edge's exploration. $\square$

3.2. MaxStarExp(k) *is APX-complete for* $k \geq 4$ *labels per edge*

It can be shown that the reduction of Section 3.1 linearly preserves approximability features; this would in turn prove that MaxStarExp(k) is APX-hard for $k \geq 6$, since MAX3SAT(3), i.e. the maximization version of 3SAT(3), is APX-complete [8]. However, this leaves a gap in the complexity of the problem for $k \in \{4, 5\}$. To close this gap we instead give an *L-reduction* [40] from the Max2SAT(3) problem, i.e. an approximation preserving reduction which linearly preserves approximability features. Max2SAT(3) is known to be APX-complete [40].

MAX2SAT(3)
Input: A boolean formula in CNF with variables $x_1, x_2, \ldots, x_p$ and clauses $c_1, c_2, \ldots, c_q$, such that each clause has at most 2 literals, and each variable appears in at most 3 clauses.
Output: Maximum number of satisfiable clauses in the formula.

The reduction:. Given an instance F of MAX2SAT(3) we shall create an instance (G_s, L) of MaxStarExp(k) such that F has β satisfiable clauses if and only if (G_s, L) has $\beta + 3p$ explorable edges. As previously, we assume without loss of generality that every variable appears once as a negative literal and once or twice as a positive literal.

The reduction is the same as the one presented in Section 3.1, with the edges of (G_s, L) being assigned the same labels as in the previous reduction to appropriately introduce conflicts between exploration windows of edges. The only difference in the construction is that now we start from a 2-CNF formula F (instead of a 3-CNF formula in Section 3.1). Thus every edge of (G_s, L) that corresponds to a clause of F now receives four labels instead of six, i.e. two labels for every literal that appears in the clause.

The following lemmas are needed for the proof of APX-hardness (Theorem 3). The proof of Lemma 3 is similar to the proof of Lemma 1, with the difference that clauses of F are of the form $c_j = (l_\pi \vee l_\rho)$ rather than

$c_j = (l_\pi \vee l_\rho \vee l_\sigma)$. The proof of Lemma 4 is exactly the same as the proof of Lemma 2.

Lemma 3. *There exists a (partial) exploration J of (G_s, L) of maximum size which explores all the edges e_i, e_i', e_i'', $i = 1, 2, \ldots, p$.*

Lemma 4. *There exists a truth assignment τ of F with $|\tau(F)| \geq \beta$ if and only if there exists a (partial) exploration J of (G_s, L) of size $|J| \geq 3p + \beta$.*

Using Lemma 4 we can now prove the APX-hardness of MaxStarExp(k).

Theorem 3. MaxStarExp(k) *is APX-hard, for $k \geq 4$.*

Proof. Denote by $OPT_{\text{Max2SAT(3)}}(F)$ the greatest number of clauses that can be simultaneously satisfied by a truth assignment of F. Also, denote by $OPT_{\text{MaxStarExp}}((G_s, L))$ the greatest number of edges that can be explored by an exploration of (G_s, L). The proof is done by an *L-reduction* [40] from the Max2SAT(3) problem, i.e. by an approximation preserving reduction which linearly preserves approximability features. For such a reduction, it suffices to provide a polynomial-time computable function g and two constants $\gamma, \delta > 0$ such that:

- $OPT_{\text{MaxStarExp}}((G_s, L)) \leq \gamma \cdot OPT_{\text{Max2SAT(3)}}(F)$, for any boolean formula F, and
- for any (partial) exploration J' of (G_s, L), $g(J')$ is a truth assignment for F and $OPT_{\text{Max2SAT(3)}}(F) - |g(J')| \leq \delta(OPT_{\text{MaxStarExp}}((G_s, L)) - |J'|)$, where $|g(J')|$ is the number of clauses of F that are satisfied by $g(J')$.

We will prove the first condition for $\gamma = 13$. Note that a random truth assignment satisfies each clause of F with probability at least $\frac{1}{2}$ (if each clause had exactly 2 literals, then it would be satisfied with probability $\frac{3}{4}$, but we have to account also for single-literal clauses), and thus there exists an assignment τ that satisfies at least $\frac{q}{2}$ clauses of F. Furthermore, since every clause has at most 2 literals and every variable appears at least once, it follows that $q \geq \frac{p}{2}$. Therefore $OPT_{\text{Max2SAT(3)}}(F) \geq \frac{q}{2} \geq \frac{p}{4}$, and thus $p \leq 4 \cdot OPT_{\text{Max2SAT(3)}}(F)$. Now Lemma 4 implies that:

$$\begin{aligned} OPT_{\text{MaxStarExp}}((G_s, L)) &= 3p + OPT_{MAX2SAT(3)}(F) \\ &\leq 3 \cdot 4 \cdot OPT_{MAX2SAT(3)}(F) + OPT_{MAX2SAT(3)}(F) \\ &= 13 \cdot OPT_{\text{Max2SAT(3)}}(F) \end{aligned}$$

To prove the second condition for $\delta = 1$, consider an arbitrary partial exploration J' of $G_s(L)$. As described in the ($\Leftarrow$)-part of the proof of Lemma 4 (same as the ($\Leftarrow$)-part of the proof of Lemma 2), we construct in polynomial time a truth assignment $g(J') = \tau$ that satisfies at least $OPT_{\text{MaxStarExp}}((G_s, L)) - 3p$ clauses of F, i.e. $|g(J')| = |\tau(F)| \geq |J'| - 3p$. Then:

$$\begin{aligned} OPT_{\text{Max2SAT(3)}}(F) - |g(J')| &\leq OPT_{\text{Max2SAT(3)}}(F) - |J'| + 3p \\ &= OPT_{\text{MaxStarExp}}((G_s, L)) - 3p - |J'| + 3p \\ &= OPT_{\text{MaxStarExp}}((G_s, L)) - |J'| \end{aligned}$$

This completes the proof of the theorem. □

Corollary 1. MaxStarExp(k) *is APX-complete, for* $k \geq 4$.

Proof. MaxStarExp(k) is a special case of the Job Interval Selection Problem where every job has at most $k - 1$ intervals[6] . To verify this, observe that every edge e with labels $l_1, l_2, \ldots, l_k$ can be seen as a job to be scheduled where the corresponding intervals are $[l_1, l_2], [l_2, l_3], \ldots, [l_{k-1}, l_k]$.

Therefore, the known (1/1.582)-approximation algorithm for the Job Interval Selection Problem with bounded number of intervals per job [18] directly implies a (1/1.582)-approximation for MaxStarExp(k). The latter, combined with the APX-hardness of MaxStarExp(k), concludes the proof. □

Now we prove a correlation between the inapproximability bounds for the MaxStarExp(k) problem and Max2SAT(3), as a result of the L-reduction presented in the proof of Theorem 3. Note that, since Max2SAT(3) is APX-hard [8], there exists a constant $\varepsilon_0 > 0$ such that there exists no polynomial-time constant-factor approximation algorithm for Max2SAT(3) with approximation ratio greater than $(1 - \varepsilon_0)$, unless P = NP.

[6]For a detailed definition of the Job Interval Selection Problem see Section 1.

Theorem 4. *Let $\varepsilon_0 > 0$ be the constant such that, unless $P = NP$, there exists no polynomial-time constant-factor approximation algorithm for* MAX2SAT(3) *with approximation ratio greater than $(1 - \varepsilon_0)$. Then, unless $P = NP$, there exists no polynomial-time constant-factor approximation algorithm for* MAXSTAREXP(k) *with approximation ratio greater than $(1 - \frac{\varepsilon_0}{13})$.*

Proof. Let $\varepsilon > 0$ be a constant such that there exists a polynomial-time approximation algorithm $\mathcal{A}$ for MAXSTAREXP(k) with ratio $(1-\varepsilon)$. Let F be an instance of MAX2SAT(3) with p variables and q clauses. We construct the instance (G_s, L) of MAXSTAREXP(k) corresponding to F, as described in the L-reduction (see Theorem 3). Then we apply the approximation algorithm $\mathcal{A}$ to (G_s, L), which returns a (partial) exploration J. Note that $|J| \geq (1-\varepsilon) \cdot OPT_{\text{MaxStarExp}}$. As described in the proof of Lemma 2, we construct from J in polynomial time a truth assignment τ; we denote by $|\tau|$ the number of clauses in F that are satisfied by the truth assignment τ. It now follows from the proof of Theorem 3 that:

$$\begin{aligned} OPT_{\text{Max2SAT(3)}}(F) - |\tau| &\leq OPT_{\text{MaxStarExp}}((G_s, L)) - |J| \\ &\leq 13\varepsilon \cdot OPT_{\text{Max2SAT(3)}}(F) \end{aligned}$$

Therefore $|\tau| \geq (1 - 13\varepsilon) \cdot OPT_{\text{Max2SAT(3)}}(F)$. That is, using algorithm $\mathcal{A}$, we can devise a polynomial-time algorithm for MAX2SAT(3) with approximation ratio $(1 - 13\varepsilon)$. Therefore, due to the assumptions of the theorem it follows that $\varepsilon \geq \frac{\varepsilon_0}{13}$, unless P = NP. This completes the proof of the theorem. ☐

Note that we have fully characterized MAXSTAREXP(k) in terms of complexity, for all values of $k \in \mathbb{N}$. However, the reduction that shows APX-hardness for MAXSTAREXP(k) cannot be employed to show NP-hardness of the decision version STAREXP(k), since the the decision problem 2SAT is polynomially solvable.

Open Problem. *What is the complexity of* STAREXP(k)*, for $k \in \{4, 5\}$?*

4. *k* random labels per edge

We now study the problem of star exploration in a temporal star graph on an underlying star graph G_s of n vertices, where the labels are assigned to the edges of G_s at random. In particular, each edge of G_s receives k labels independently of other edges, and each label is chosen uniformly at random and independently from a set of available labels. We will distinguish between two models: the "integer labels" model, where the labels are integer numbers chosen from the set of positive integers up to a lifetime $\alpha \in \mathbb{N}$, and the "continuous $[0,1]$ model", where the labels are real numbers from 0 up to 1.

In particular, for the integer labels model we give a lower and an upper bound for the number of labels per edge needed for a full exploration of a star of n vertices to be likely, for different values of the lifetime α. For the continuous $[0,1]$ model, we provide a lower bound for the number of labels per edge needed for full exploration of the star with high probability. Note that in the continuous $[0,1]$ model, the chosen labels on all edges of the underlying star graph are distinct with probability 1. Thus, any such lower bound shown for the continuous $[0,1]$ model is also a lower bound for the integer labels model (for any value of the lifetime). Conversely, any upper bound on the number of labels per edge needed for exploration of a temporal star with high probability in the integer labels model (for some value of the lifetime) is also an upper bound for the exploration of a temporal star with high probability in the continuous $[0,1]$ model.

4.1. The case of Integer labels

In this section, we assume that each label of an edge of G_s is chosen uniformly at random and independently of others from the set of integers $\{1, 2, \ldots, \alpha\}$, for some $\alpha \in \mathbb{N}$. We call this a *uniform random temporal star* with *lifetime* α and *edge availability* k, and denote it by $G_s(\alpha, k)$. In this section, we investigate the probability of exploring all edges in a uniform random temporal star based on different values of α and k, thus partially characterizing uniform random temporal stars that can be *fully* explored or not, asymptotically almost surely. In particular, as a warm-up for the results shown in the next section, we now introduce a technique, which involves pairs of edges that cannot both be explored in a full exploration of a temporal star, in order to get a lower bound on the number of labels per edge needed for full exploration of the temporal star with high probability. In Section 4.2, we generalize this technique and derive a *stronger bound* for the continuous $[0,1]$ model, which also holds for the integer labels model for any lifetime.

We introduce the following definition, needed for the proof of Theorem 5.

Definition 5 (Blocking pair of edges)**.** *Let e_1, e_2 be two edges of a uniform random temporal star $G_s(\alpha, 2)$, $\alpha \geq 2$. Let the labels of e_1 be a_1, a_2, with*

$a_1 \leq a_2$. Let the labels of e_2 be b_1, b_2, with $b_1 \leq b_2$. We say that e_1, e_2 are a blocking pair *(with respect to exploration in $G_s(\alpha, 2)$) if $a_1 \leq b_1 \leq a_2 \leq b_2$, or $a_1 \leq b_1 \leq b_2 \leq a_2$, or $b_1 \leq a_1 \leq b_2 \leq a_2$, or $b_1 \leq a_1 \leq a_2 \leq b_2$.*

Theorem 5. *If $\alpha \geq 4$ and $k = 2$, then the probability that we can explore all edges of $G_s(\alpha, k)$ tends to zero as n tends to infinity.*

Proof. Consider two particular edges e_1, e_2 of $G_s(\alpha, 2)$, $\alpha \geq 4$. Let $\mathcal{E}$ be the event that e_1, e_2 are a blocking pair and $\mathcal{E}'$ be the event that e_1, e_2 have 4 distinct labels in total. Then, the probability that e_1, e_2 are a blocking pair is:

$$Pr[\mathcal{E}] = Pr[\mathcal{E} \mid \mathcal{E}'] \cdot Pr[\mathcal{E}'] + Pr[\mathcal{E} \mid \bar{\mathcal{E}'}] \cdot Pr[\bar{\mathcal{E}'}] \geq Pr[\mathcal{E} \mid \mathcal{E}'] \cdot Pr[\mathcal{E}'], \quad (1)$$

where $\bar{\mathcal{E}'}$ denotes the complement of $\mathcal{E}'$. Note that if all labels of e_1, e_2 are distinct, then the probability that e_1, e_2 are a blocking pair is exactly the ratio of the "good" arrangements of the 4 distinct labels, i.e. those where $a_1 < b_1 < a_2 < b_2$, or $a_1 < b_1 < b_2 < a_2$, or $b_1 < a_1 < b_2 < a_2$, or $b_1 < a_1 < a_2 < b_2$, over the total number of possible arrangements of the 4 distinct labels (recall that, by our naming convention, $a_1 < a_2$ and $b_1 < b_2$ always hold so the number of such arrangements is 6 in total). So, equation 1 becomes:

$$Pr[\mathcal{E}] \geq \frac{4}{6} \cdot Pr[\mathcal{E}']. \quad (2)$$

Now, the probability that all 4 labels of e_1, e_2 are distinct is:

$$Pr[\mathcal{E}'] = \left(1 - \frac{1}{\alpha}\right) \cdot \left(1 - \frac{2}{\alpha}\right) \cdot \left(1 - \frac{3}{\alpha}\right) \geq \frac{3}{4} \cdot \frac{2}{4} \cdot \frac{1}{4} = \frac{3}{32}. \quad (3)$$

Therefore, by equation 3, equation 2 becomes $Pr[\mathcal{E}] \geq \frac{1}{16}$.
So, we have:

$$Pr[e_1, e_2 \text{ are not a blocking pair}] \leq \frac{15}{16}.$$

We will now *partition* the edges into pairs (rather than consider *all possible pairs* of edges) to ensure independence and simplify the calculations that follow. So, we consider an arbitrary partition of all edges of $G_s(\alpha, 2)$ into $\lfloor \frac{n-1}{2} \rfloor$ independent pairs (with the possibility of an edge remaining unpaired). If there is an exploration in $G_s(\alpha, 2)$, then there are no blocking pairs of edges in any such pairing and, thus, in the particular pairing we have chosen. Therefore, the probability that we can explore all edges is:

$$
\begin{aligned}
Pr[\text{exploration}] &\leq Pr[\text{no blocking pair exists in the pairing}] \\
&\leq \left(\frac{15}{16}\right)^{\lfloor \frac{n-1}{2} \rfloor} \to 0, \text{ as } n \to +\infty
\end{aligned}
$$

□

Theorem 6. *If $\alpha \geq 2(n-1)$ and $k \geq 12n \ln n$, then the probability that we can explore all edges of $G_s(\alpha, k)$ tends to 1 as n tends to infinity.*

Proof. We consider the time-line from 1 to α and we split it into $2(n-1)$ consecutive time-windows, each of which, excluding possibly the last time-window, is of size $\lfloor \frac{\alpha}{2(n-1)} \rfloor$ as shown in Figure 3; the last time window may be of larger size if α is not divisible by $2(n-1)$. Let us henceforth refer to those time-windows as *boxes* and denote the i^{th} such box by B_i. The first box contains the labels $1, 2, \ldots, \lfloor \frac{\alpha}{2(n-1)} \rfloor$, the second box contains the labels $\lfloor \frac{\alpha}{2(n-1)} \rfloor + 1, \lfloor \frac{\alpha}{2(n-1)} \rfloor + 2, \ldots, 2\lfloor \frac{\alpha}{2(n-1)} \rfloor$, and so on.

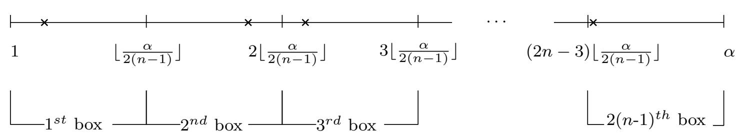


Figure 3: Splitting the time from 1 to α into $2(n-1)$ boxes to show the existence of at least one label per box, for every edge, asymptotically almost surely.

We will show that for every edge of G_s, there will be asymptotically almost surely at least one of its labels that falls in the first box, one of its labels that falls in the second box, etc. But first, let us note the following:

Observation 1. *If for every edge $e \in E$ and for every box B_i there is at least one label of e that lies within B_i, then there exists an exploration of $G_s(\alpha, k)$.*

Proof of Observation. Assume that for every edge $e \in E$ and for every box B_i there is at least one label of e that lies within B_i. Fix an arbitrary order $e_1, e_2, \ldots, e_{n-1}$ of the edges of G_s. We can fully explore G_s, by exploring every e_i, $i = 1, \ldots, n-1$, in the above order using the label of e_i that lies within B_{2i-1} to enter e_i, and the label of e_i that lies within B_{2i} to exit. This completes the proof of Observation 1 . ■

Note now that for a particular edge $e \in E$ and a particular box B_i of e, the probability that B_i contains none of the labels of e is:

$$Pr[B_i \text{ is empty}] = \left(1 - \frac{\left\lfloor \frac{\alpha}{2(n-1)} \right\rfloor}{\alpha}\right)^k \leq \left(1 - \frac{\frac{1}{2} \cdot \frac{\alpha}{2(n-1)}}{\alpha}\right)^{12n \ln n} \leq \frac{1}{n^3}.$$

So, by the union bound, the probability that there is an empty box of e is:

$$Pr[\text{there is an empty } B_i \text{ of } e] \leq \#\text{boxes} \cdot \frac{1}{n^3} = 2(n-1) \cdot \frac{1}{n^3} \leq \frac{2}{n^2},$$

and so the probability that there exists an edge with an empty box is, again by the union bound:

$$Pr[\text{there is an edge with an empty box}] \leq \#\text{edges} \cdot \frac{2}{n^2} \leq \frac{2}{n}.$$

Finally, the probability that we can explore all edges of $G_s(\alpha, k)$ is:

$$Pr[\text{exploration}] \geq 1 - \frac{2}{n} \to 1, \text{ as } n \to +\infty$$

The latter completes the proof of Theorem 6. □

Figure 4 shows the current state of what is known for the explorability of $G_s(\alpha, k)$ depending on the values of the lifetime, α, and the edge availability, k. Notice that any exploration of a star graph G_s needs at least $2(n-1)$ distinct labels to explore all $n-1$ edges, hence any graph with $\alpha < 2(n-1)$ is not explorable.

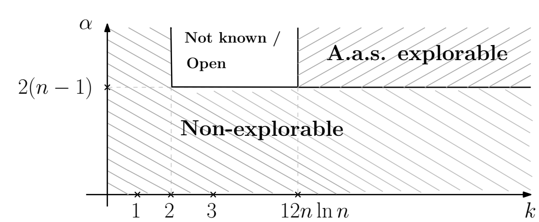


Figure 4: The shaded areas of the chart indicate the pairs (α, k) for which $G_s(\alpha, k)$ is asymptotically almost surely (a.a.s.) explorable and non-explorable, respectively.

4.2. The case of labels in the continuous interval [0,1]

We consider now the explorability of a temporal version of the star graph G_s with n vertices, in which each edge selects k time labels independently and uniformly at random from the interval $[0, 1]$, for some integer $k > 0$ (k may depend on n). We denote this model as $G_s([0, 1], k)$. In this model, the $k(n-1)$ chosen labels on all edges of G_s are distinct with probability 1, and so explorability of the graph depends only on the relative ordering of the labels; we will refer to a relative ordering σ of the labels as a *(label) configuration*. In fact, without loss of generality, we view this ordering as a permutation of $k(n-1)$ items which belong to $n-1$ distinct groups (corresponding to the edges) of k identical items each. Therefore, there are $\binom{k(n-1)}{k,\ldots,k} = \frac{(k(n-1))!}{(k!)^{n-1}}$ different configurations. For example, consider the star G_s with $n = 4$ vertices and 3 edges, e_1, e_2, e_3; a possible configuration for $G_s([0, 1], 2)$ is $(e_1, e_2, e_3, e_1, e_2, e_3)$ (for which we do not have explorability). Finally, we will denote by $G_s([0, 1], k|\sigma)$ the instance of $G_s([0, 1], k)$ that has configuration σ. If $G_s([0, 1], k|\sigma)$ is not explorable, we say that σ is a *blocking configuration.*

We first show the following:

Lemma 5. *When $k = n-1$, there is a blocking configuration σ for $G_s([0, 1], k)$.*

Proof. For each $i \in [n-1]$ we let $\sigma[j] = e_{i+1}$ iff $(j \mod n-1) = i,\ j \leq k(n-1)$. Alternatively, σ consists of $k = n-1$ concatenations of the sequence $e_1, e_2, \ldots, e_{n-1}$. Notice now that, between any two labels of the same edge there is at least one label from each of the other edges. Therefore, after we have explored $n-2$ edges, the last edge to be explored will have at most $k-(n-2) = 1$ available labels, which is not enough to explore it (remember exploration means that we start and finish at the center of the star). □

We now give a lower bound on the probability that we get a blocking configuration when the number of edges is equal to the number of labels per edge:

Lemma 6. *Let the number of edges of G_s be $n-1 = k$ and let σ be a randomly chosen configuration for $G_s([0, 1], k)$. Then* $\Pr(\sigma$ *is blocking*$) \geq \frac{(k!)^{k+1}}{(k^2)!} \geq k^{-k^2}$.

Proof. First note that (as mentioned also above) the total number of distinct label configurations of $G_s([0,1],k)$ equals $\binom{k^2}{k,\ldots,k} = \frac{(k^2)!}{(k!)^k}$. Second, in view of the construction of the blocking configuration in Lemma 5, we get distinct blocking configurations for different permutations of the k edges. Therefore, the probability that we get a blocking configuration is at least $\frac{(k!)^{k+1}}{(k^2)!}$.

To show that $\frac{(k!)^{k+1}}{(k^2)!} \geq k^{-k^2}$, we use the following bound on $m!$, for any integer m, namely $\sqrt{2\pi} m^{m+\frac{1}{2}} e^{-m} \leq m! \leq e m^{m+\frac{1}{2}} e^{-m}$. In particular, we get

$$\begin{aligned}
\frac{(k!)^{k+1}}{(k^2)!} &\geq \frac{\left(\sqrt{2\pi} k^{k+\frac{1}{2}} e^{-k}\right)^{k+1}}{e(k^2)^{k^2+\frac{1}{2}} e^{-k^2}} \\
&= \frac{(\sqrt{2\pi})^{k+1} k^{k^2+\frac{3}{2}k+\frac{1}{2}} e^{-k^2-k}}{k^{2k^2+1} e^{-k^2+1}} \\
&= (\sqrt{2\pi})^{k+1} k^{-k^2+\frac{3}{2}k-\frac{1}{2}} e^{-k-1}.
\end{aligned}$$

For any $k \geq 2$, the above is at least k^{-k^2}, as needed. Finally, we note that for $k = 1$ the bound holds trivially.

$\square$

Theorem 7. *Let σ be a randomly chosen configuration for $G_s([0,1],k)$. Then $\Pr(G_s([0,1],k|\sigma)$ is explorable$) \leq \left(1 - k^{-k^2}\right)^{\frac{n-k}{k}}$. In particular, for any $k \leq (\ln n)^{\frac{1}{2}-\epsilon}$, where $\epsilon > 0$ is a constant, $G_s([0,1],k)$ is not explorable with high probability.*

Proof. For a set of k edges of G_s inducing a star H, let $\sigma_{|H}$ denote the part of σ that concerns the relative ordering of labels of edges in H. In particular, if $\sigma_{|H}$ is blocking for $H([0,1],k)$, then σ is blocking for $G_s([0,1],k)$.

In view of this, we partition the set of edges of G_s into $\lceil \frac{n-1}{k} \rceil$ sets inducing (edge disjoint) stars $H_1, \ldots, H_{\lceil \frac{n-1}{k} \rceil}$, having exactly k edges each, except maybe for the last one. For $G_s([0,1],k|\sigma)$ to be explorable, each of $H_i([0,1],k|\sigma_{|H_i})$ should be explorable, for $i = 1, 2, \ldots, \lceil \frac{n-1}{k} \rceil$. Since label choices of disjoint sets of edges are independent (and thus the events of explorability for distinct H_i's are independent), using the upper bound $1 - k^{-k^2}$

for the explorability of the star with k edges, we get

$$\begin{aligned}
\Pr(G_s([0,1],k|\sigma) \text{ is explorable}) &\leq \prod_{i=1}^{\left\lceil \frac{n-1}{k} \right\rceil} \Pr(H_i([0,1],k|\sigma_{|H_i}) \text{ is explorable}) \\
&\leq \left(1 - k^{-k^2}\right)^{\left\lfloor \frac{n-1}{k} \right\rfloor} \\
&\leq \left(1 - k^{-k^2}\right)^{\frac{n-k}{k}} \leq e^{-\frac{n-k}{k}k^{-k^2}},
\end{aligned}$$

where in the third inequality we used the fact that $\left\lfloor \frac{n-1}{k} \right\rfloor \geq \frac{n-k}{k}$, for any $1 \leq k \leq n$. It is now straightforward to verify that $e^{-\frac{n-k}{k}k^{-k^2}}$ tends to 0 for $k \leq (\ln n)^{\frac{1}{2}-\epsilon}$, for any positive constant ϵ.

□

Note: The above bound could potentially be improved by considering all $\binom{n-1}{k}$ subsets of k edges of G_s (with $n-1$ total edges), but the corresponding events are no longer independent, and so the analysis should be much harder.

5. Conclusions and open problems

In this paper, we have thoroughly investigated the computational complexity landscape of the temporal star exploration problems StarExp(k) and MaxStarExp(k), depending on the maximum number k of labels allowed per edge.

We have shown that an optimal solution to the maximization problem, on instances every edge of which has two or three labels, can be efficiently found in $O(n \log n)$ time. This immediately implies that the decision version can be also solved in the same time. We show that StarExp(k) is NP-complete, for every $k \geq 6$, and MaxStarExp(k) is APX-complete, for every $k \geq 4$. Finally, we study the problem of exploring uniform random temporal stars whose edges have k random labels (chosen uniformly at random within an interval $[1, \alpha]$, for some $\alpha \in \mathbb{N}$). We partially characterize the classes of uniform random temporal stars which, asymptotically almost surely, admit a complete (resp. admit no complete) exploration. In particular, the "blocking pairs" technique used to show that there is asymptotically almost surely no complete exploration for $k = 2$ and $\alpha \geq 4$ cannot be easily extended to large k. So, it remains open to determine the explorability of uniform random temporal stars for values of k between 2 and $6n \ln n$.

We pose here a question regarding the complexity of StarExp(k), for $k \in \{4, 5\}$, which still remains unknown. An interesting variation of StarExp(k) and MaxStarExp(k) is the case where the consecutive labels of every edge are λ time steps apart, for some $\lambda \in \mathbb{N}$. What is the complexity and/or best approximation factor one may hope for in this case?

6. Acknowledgments

This work was supported in part by:

(i) the NeST initiative of the EEE/CS School of the University of Liverpool,

(ii) the EPSRC grant on "Algorithmic Aspects of Temporal Graphs" EP/P020372/1,

(iii) the EPSRC grant on "Algorithmic Aspects of Temporal Graphs" EP/P02002X/1, and

(iv) the EPSRC AlgoUK network grant EP/R005613/1.

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

[1] E. Aaron, D. Krizanc, and E. Meyerson. DMVP: foremost waypoint coverage of time-varying graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 29–41, 2014.

[2] E. C. Akrida, J. Czyzowicz, L. Gasieniec, L. Kuszner, and P. G. Spirakis. Temporal flows in temporal networks. *Journal of Computer and System Sciences*, 103:46–60, 2019.

[3] E. C. Akrida, L. Gasieniec, G. B. Mertzios, and P. G. Spirakis. Ephemeral networks with random availability of links: The case of fast networks. *Journal of Parallel and Distributed Computing*, 87:109–120, 2016.

[4] E. C. Akrida, L.Gasieniec, G. B. Mertzios, and P. G. Spirakis. The complexity of optimal design of temporally connected graphs. *Theory of Computing Systems*, 61(3):907–944, 2017.

[5] E. C. Akrida, G. Mertzios, P. G. Spirakis, and V. Zamaraev. Temporal vertex covers and sliding time windows. *Journal of Computer and System Sciences*, 107:108–123, 2020.

[6] E. C. Akrida and P. G. Spirakis. On verifying and maintaining connectivity of interval temporal networks. In *International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS)*, pages 142–154, 2015.

[7] A. Asadpour, M. X. Goemans, A. Madry, S. O. Gharan, and A. Saberi. An $O(\log\ n/\log\ \log\ n)$-approximation algorithm for the asymmetric traveling salesman problem. *Operations Research*, 65, 2017.

[8] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer-Verlag, 1st edition, 1999.

[9] Y. Azar and A. Vardi. Dynamic traveling repair problem with an arbitrary time window. In *International Workshop on Approximation and Online Algorithms (WAOA)*, pages 14–26, 2016.

[10] S. Biswas, A. Ganguly, and R. Shah. Restricted shortest path in temporal graphs. In *International Conference on Database and Expert Systems Applications (DEXA)*, 2015.

[11] H. L. Bodlaender and T. C. van der Zanden. On exploring temporal graphs of small pathwidth. *CoRR*, abs/1807.11869, 2018.

[12] B.-M. Bui-Xuan, A. Ferreira, and A. Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(2):267–285, 2003.

[13] A. Casteigts and P. Flocchini. Deterministic Algorithms in Dynamic Networks: Formal Models and Metrics. Technical report, Defence R&D Canada, 2013.

[14] A. Casteigts and P. Flocchini. Deterministic Algorithms in Dynamic Networks: Problems, Analysis, and Algorithmic Tools. Technical report, Defence R&D Canada, 2013.

[15] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems (IJPEDS)*, 27(5):387–408, 2012.

[16] T. H. Chan and L. Ning. Fast convergence for consensus in dynamic networks. *ACM Transactions on Algorithms*, 10, 2014.

[17] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[18] J. Chuzhoy, R. Ostrovsky, and Y. Rabani. Approximation algorithms for the Job Interval Selection Problem and related scheduling problems. *Mathematics of Operations Research*, pages 730–738, 2006.

[19] A. E. F. Clementi, C. Macci, A. Monti, F. Pasquale, and R. Silvestri. Flooding time of Edge-Markovian evolving graphs. *SIAM Journal on Discrete Mathematics (SIDMA)*, 24(4):1694–1712, 2010.

[20] M. de Berg, K. Buchin, B. M. P. Jansen, and G. J. Woeginger. Fine-grained complexity analysis of two classic TSP variants. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, 2016.

[21] V. G. Deineko and G. J. Woeginger. Another look at the shoelace TSP: the case of very old shoes. In *International Conference on Fun with Algorithms (FUN)*, 2014.

[22] C. Demetrescu and G. F. Italiano. Algorithmic techniques for maintaining shortest routes in dynamic networks. *Electronic Notes in Theoretical Computer Science*, 171, 2007.

[23] T. Erlebach, M. Hoffmann, and F. Kammer. On temporal graph exploration. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 444–455, 2015.

[24] A. Ferreira. Building a reference combinatorial model for MANETs. *IEEE Network*, 18:24–29, 2004.

[25] P. Flocchini, B. Mans, and N. Santoro. Exploration of periodically varying graphs. In *International Symposium on Algorithms and Computation (ISAAC)*, pages 534–543, 2009.

[26] A. M. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12, 1982.

[27] S. O. Gharan, A. Saberi, and M. Singh. A randomized rounding approach to the traveling salesman problem. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 550–559, 2011.

[28] G. Giakkoupis, T. Sauerwald, and A. Stauffer. Randomized rumor spreading in dynamic graphs. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 495–507, 2014.

[29] G. Hadjicharalambous, P. C. Pop, E. Pyrga, G. Tsaggouris, and C. D. Zaroliagis. The railway traveling salesman problem. In *International Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS)*, 2004.

[30] A. Himmel, H. Molter, R. Niedermeier, and M. 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.

[31] D. Ilcinkas, R. Klasing, and A. M. Wade. Exploration of constantly connected dynamic graphs based on cactuses. In *International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, 2014.

[32] D. Kempe, J. M. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. In *ACM symposium on Theory of computing (STOC)*, pages 504–513, 2000.

[33] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison-Wesley Longman, 2005.

[34] O. Kostakis and A. Gionis. On mining temporal patterns in dynamic graphs, and other unrelated problems. In *International Conference on Complex Networks and Their Applications (COMPLEX NETWORKS)*, pages 516–527, 2017.

[35] O. Kostakis, N. Tatti, and A. Gionis. Discovering recurring activity in temporal networks. *Data Mining and Knowledge Discovery*, 31(6):1840–1871, 2017.

[36] J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1, 2007.

[37] G. B. Mertzios, O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Temporal network optimization subject to connectivity constraints. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 657–668, 2013.

[38] O. Michail and P. G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical Computer Science*, 634:1–23, 2016.

[39] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[40] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43(3):425–440, 1991.

[41] N. Paulsen, F. Diedrich, and K. Jansen. Heuristic Approaches to minimize tour duration for the TSP with multiple time windows. In *Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)*, pages 42–55, 2015.

[42] A. Sebö and J. Vygen. Shorter tours by nicer ears: 7/5-Approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, 34, 2014.

[43] F. C. R. Spieksma. On the approximability of an interval scheduling problem. *Journal of Scheduling*, pages 215–227, 1999.

[44] O. Svensson, J. Tarnawski, and L. A. Végh. A constant-factor approximation algorithm for the asymmetric traveling salesman problem. In *Symposium on Theory of Computing, STOC*, 2018.

[45] J. K. Tang, M. Musolesi, C. Mascolo, and V. Latora. Characterising temporal distance and reachability in mobile and online social networks. *ACM Computer Communication Review*, 40:118–124, 2010.

[46] J. Viard, M. Latapy, and C. Magnien. Revealing contact patterns among high-school students using maximal cliques in link streams. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1517–1522, 2015.

[47] T. Viard, M. Latapy, and C. Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609:245–252, 2016.

[48] D. Wagner, T. Willhalm, and C. D. Zaroliagis. Dynamic shortest paths containers. *Electronic Notes in Theoretical Computer Science*, 92, 2004.

[49] H. Zhuang, Y. Sun, J. Tang, J. Zhang, and X. Sun. Influence maximization in dynamic social networks. In *IEEE International Conference on Data Mining*, 2013.