Traveling Salesman Problems in Temporal Graphs[☆,☆☆]

Othon Michail[a,*], Paul G. Spirakis[a,b]

[a] *Computer Technology Institute & Press "Diophantus" (CTI), Patras, Greece*
[b] *Department of Computer Science, University of Liverpool, UK*

Abstract

In this work, we introduce the notion of time to some well-known combinatorial optimization problems. In particular, we study problems defined on *temporal graphs*. A temporal graph $D = (V, A)$ may be viewed as a time-sequence $G_1, G_2, \ldots, G_l$ of static graphs over the same (static) set of nodes V. Each $G_t = D(t) = (V, A(t))$ is called the *instance of* D *at time* t and l is called the *lifetime of* D. Our main focus is on analogues of *traveling salesman problems* in temporal graphs. A sequence of time-labeled edges (e.g. a tour) is called *temporal* if its labels are strictly increasing. We begin by considering the problem of exploring the nodes of a temporal graph as soon as possible. In contrast to the positive results known for the static case, we prove that, it cannot be approximated within cn, for some constant $c > 0$, in a special case of temporal graphs and within $(2 - \varepsilon)$, for every constant $\varepsilon > 0$, in another special case in which $D(t)$ is strongly connected for all $1 \leq t \leq l$, both unless $\mathbf{P} = \mathbf{NP}$. We then study the temporal analogue of TSP(1,2), abbreviated TTSP(1,2), where, for all $1 \leq t \leq l$, $D(t)$ is a complete weighted graph with edge-costs from $\{1, 2\}$ and the cost of an edge may vary from instance to instance. The goal is to find a minimum cost temporal TSP tour. We give several *polynomial-time approximation algorithms* for TTSP(1,2). Our best approximation is $(1.7 + \varepsilon)$ for the generic TTSP(1,2) and $(13/8 + \varepsilon)$ for its interesting special case in which the lifetime of the temporal graph is restricted to n. In the way, we also introduce temporal versions of other fundamental combinatorial optimization problems, for which we obtain polynomial-time approximation algorithms and hardness results.

Keywords:
traveling salesman problem, temporal graph, approximation algorithm, inapproximability, hardness result, dynamic network, TSP with costs one and two, temporal matching, exploration

1. Introduction

A *temporal graph* is, informally speaking, a graph that changes with time. A great variety of both modern and traditional networks such as information and communication networks, social networks, transportation networks, and several physical systems can be naturally modeled as temporal graphs. In fact, this is true for almost any network with a dynamic topology. Most modern communication networks, such as mobile ad hoc, sensor, peer-to-peer, opportunistic, and delay-tolerant networks, are inherently dynamic. In social networks, the topology usually represents the social connections between a group of individuals and it changes as the social relationships between the individuals are updated, or as individuals leave or enter the group. In a transportation network, there is usually some fixed network of routes and a set of transportation units

[☆]Supported in part by (i) the project "Foundations of Dynamic Distributed Computing Systems" (FOCUS) which is implemented under the "ARISTEIA" Action of the Operational Programme "Education and Lifelong Learning" and is co-funded by the European Union (European Social Fund) and Greek National Resources and (ii) the FET EU IP project MULTIPLEX under contract no 317532.

[☆☆]Some preliminary versions of the results in this paper have also appeared in [MS14b, Mic15].

[*]Corresponding author (Telephone number: +30 2610 960200, Fax number: +30 2610 960490, Postal Address: Computer Technology Institute & Press "Diophantus" (CTI), N. Kazantzaki Str., Patras University Campus, Rio, P.O. Box 1382, 26504, Greece).

Email addresses: michailo@cti.gr (Othon Michail), P.Spirakis@liverpool.ac.uk (Paul G. Spirakis) moving over these routes and dynamicity refers to the change of the positions of the transportation units in the network as time passes. Physical systems of interest may include several systems of interacting particles.

In this work, we restrict attention to *discrete time*. This is totally plausible when the dynamicity of the system is inherently discrete, which is for example the case in synchronous mobile distributed systems that operate in discrete rounds, but can also satisfactory approximate a wide range of continuous-time systems. Moreover, this choice gives to the resulting models and problems a purely combinatorial flavor. We also restrict attention to systems in which only the relationships between the participating entities may change and not the entities themselves. Therefore, in this paper, a temporal graph $D = (V, A)$ may always be viewed as a sequence $G_1, G_2, \ldots, G_l$ of static graphs over the same (static) set of nodes V. Each $G_t = D(t) = (V, A(t))$ is called the *instance of D at time t* and l is called the *lifetime of D*.

Though static graphs have been extensively studied, for their temporal generalization we are still far from having a concrete set of structural and algorithmic principles. Additionally, it is not yet clear how is the complexity of combinatorial optimization problems affected by introducing to them a notion of time. In an early but serious attempt to answer this question, Orlin [Orl81] observed that many dynamic languages derived from **NP**-complete languages can be shown to be **PSPACE**-complete. Among the other few things that we do know, is that the max-flow min-cut theorem holds with unit capacities for time-respecting paths [Ber96]. Additionally, Kempe *et al.* [KKK00] proved that, in temporal graphs, the classical formulation of Menger's theorem is violated and the computation of the number of node-disjoint s-t paths becomes **NP**-complete. Note that, in both [Ber96] and [KKK00] a labeled path (where labels correspond to instances of availability) is called *time-respecting* (also called *temporal* in the present work) if its labels are non-decreasing. In this work, following [MMCS13], we require a strictly increasing sequence of labels. This choice is very well motivated by recent work in dynamic communication networks. If it takes one time unit to transmit a data packet over a link then a packet can only be transmitted over paths with strictly increasing availability times. Recently, building on the distributed online dynamic network model of [KLO10], Dutta *et al.* [DPR$^+$13], among other things, presented *offline centralized algorithms* (as are the algorithms that we consider in this work) for the k-gossip problem. In k-gossip, there are k distinct pieces of information (tokens) that are initially present in some distributed processes and the problem is to disseminate all the k tokens to all the processes in the dynamic network, under the constraint that one token can go through an edge per round.

In another very recent work [MMCS13], the authors achieved a reformulation of Menger's theorem which is valid for all temporal graphs and introduced several interesting cost minimization parameters for optimal temporal network design. One is the *temporality* of a graph G, in which the goal is to create a temporal version of G minimizing the maximum number of labels of an edge, and the other is the *temporal cost* of G, in which the goal is to minimize the total number of labels used. Optimization of these parameters is performed subject to some *connectivity constraint*. They proved several upper and lower bounds for the temporality of some very basic graph families such as rings, directed acyclic graphs, and trees, as well as a trade-off between the temporality and the maximum label of rings. Furthermore, they gave a *generic method* for computing a lower bound of the temporality of an arbitrary graph w.r.t. the constraint of preserving a time-respecting analogue of every simple path of G. Finally, they proved that computing the temporal cost w.r.t. the constraint of preserving at least one time-respecting path from u to v whenever v is reachable from u in G, is **APX**-hard. For a very recent introduction to temporal graphs from an algorithmic point of view, the interested reader is referred to [Mic15].

We make here one more step towards the direction of revealing the algorithmic principles of temporal graphs. In particular, we introduce the study of traveling salesman problems in temporal graphs, which, to the best of our knowledge, have not been considered before in the literature. Our main focus is on the Temporal Traveling Salesman Problem with Costs One and Two abbreviated TTSP(1,2) throughout the paper. In this problem, we are given a temporal graph $D = (V, A)$ every instance of which is a complete graph, i.e. $D(t) = (V, A(t))$ is complete for all $1 \leq t \leq l$. Moreover, the edges of every $D(t)$ are weighted according to a cost function $c : A \to \{1, 2\}$. Observe that A is a set of *time-edges* (e, t), where e is an edge and t is the time at which e appears. So, the cost function c is allowed to assign different cost values to different instances of the same edge, therefore, in this model, *the costs are dynamic* in nature (see Figure 1 for an illustration). We are asked to find a *Temporal TSP tour* (abbreviated *TTSP tour*) of minimum total cost. A TSP tour $(u_1, t_1, u_2, t_2, \ldots, t_{n-1}, u_n, t_n, u_1)$ is temporal if $t_i < t_{i+1}$ for all $1 \le i \le n-1$. The cost of such a TSP tour is $\sum_{1 \le i \le n} c((u_i, u_{i+1}), t_i)$, where $u_{n+1} = u_1$. Additionally, note that, in general, we do not make any assumptions about the lifetime of D. Of course, for a temporal TSP tour to exist the lifetime has to be at least $n-1$, because the tour can visit at most one new node at a time, however, in general, it can be much greater than n. Whenever we restrict attention to limited lifetime, this will be explicitly stated. We should also emphasize that, throughout this work, *the entire temporal graph is provided to the centralized algorithms in advance*. For example, to provide the temporal graph of Figure 1 as input to an algorithm we have to provide explicitly all six instances of the temporal graph. It is useful to observe that TTSP(1,2) seems to be naturally closer to the ASYMMETRIC TSP(1,2) and not to the SYMMETRIC TSP, and this seems to hold independent of whether the temporal graph has directed or undirected instances. The reason is that dynamic costs, no matter whether they occur over directed or undirected edges, are asymmetric in nature. Requiring an edge, say undirected, of the temporal graph to be symmetric would mean that if the edge has cost k at time 1 then it should also have cost k at all (or at least some) subsequent times, which would bring the model much closer to a static one. Throughout the work we assume directed edge-sets, however keep in mind that, as argued above, the undirected case is not expected to be any simpler. Finally, note that ATSP(1,2) (standing for Asymmetric TSP) is a special case of TTSP(1,2) when the lifetime is restricted to n and $c(e,t) = c(e,t')$ for all edges e and times t, t'. This immediately implies that TTSP(1,2) is also **APX**-hard [PY93] and cannot be approximated within any factor less than 207/206 [KS13] (and the same holds for the special case of TTSP(1,2) with lifetime restricted to n that we also consider in this work).

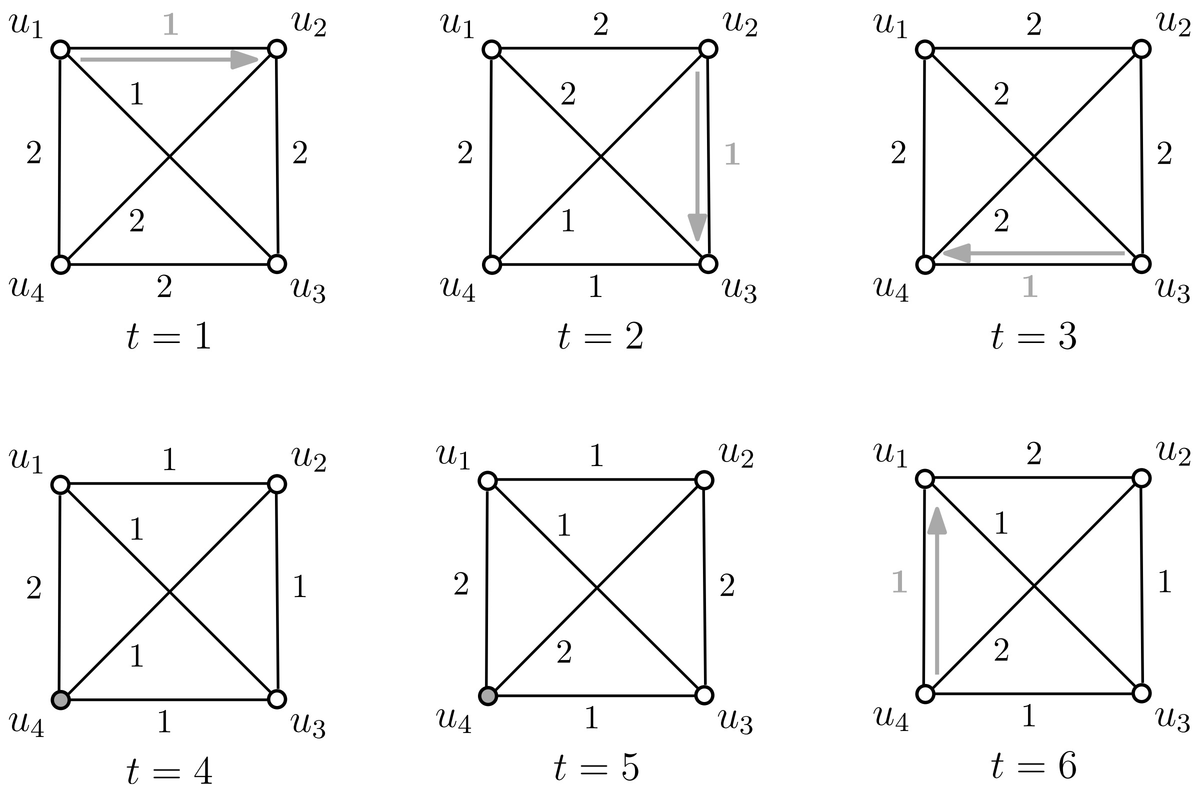

Figure 1: An instance of TTSP(1,2) consisting of a complete temporal graph $D = (V, A)$, where $V = \{u_1, u_2, u_3, u_4\}$, and a cost function $c : A \to \{1, 2\}$ which is presented by the corresponding costs on the edges. For simplicity, D is an undirected temporal graph. Observe that the cost of an edge may change many times, e.g. the cost of u_2u_3 changes 5 times while of u_1u_4 changes only once. Here, the lifetime of the temporal graph is 6 and it is greater than $|V|$. The gray arcs and the nodes filled gray (meaning that the tour does not make a move and remains on the same node for that step) represent the TTSP tour $(u_1, 1, u_2, 2, u_3, 3, u_4, 6, u_1)$ that has cost $4 = |V|$ and therefore it is an optimum TTSP tour.

1.1. Our Approach-Contribution

We now summarize our approach to approximate TTSP(1,2). Note that all the approximation algorithms that we present in this work are *polynomial-time algorithms* on a binary encoding of the temporal graph, i.e. on $|\langle D\rangle|$. In the static case, one easily obtains a (3/2)-factor approximation for ATSP(1,2) by computing a perfect matching maximizing the number of ones and then patching the edges together arbitrarily. This works well, because such a minimum cost perfect matching can be computed in polynomial time in the static case by Edmond's algorithm [Edm65] and its cost is at most half the cost of an optimum TSP tour, as the latter consists of two perfect matchings. The 3/2 factor follows because the remaining $n/2$ edges that are added during the patching process cost at most n, which in turn is another lower bound to the cost of the optimum TSP tour. This was one of the first algorithms known for ATSP(1,2). Other approaches have improved the factor to the best currently known 5/4 [Blä04]. Unfortunately, as we shall see, even the apparently simple task of computing a matching maximizing the number of ones is not that easy in temporal graphs. In particular, we prove in Section 4.1 that computing a matching maximizing the number of ones and additionally satisfying the temporal condition that all its edges appear at distinct times is **NP**-hard. The reason that we insist on distinct times is that we can form a temporal TSP tour by patching the edges of a matching only if the edges of the matching can be strictly ordered in time. In fact, an additional requirement is that the edges of the matching should have time differences of at least two, otherwise the patching-edges would share times with some of the matching-edges and the resulting tour would not have strictly increasing labels, i.e. it would not be temporal.

We naturally then search for good approximations for the temporal matching problem. We follow two main approaches. One is to reduce the problem to the Maximum Independent Set in $(k+1)$-claw free graphs and the other is to reduce it to k'-Set Packing for some k and k' to be determined. The first approach gives a $(7/4+\varepsilon)$-approximation ($= 1.75+\varepsilon$) for the generic TTSP(1,2) and a $(12/7+\varepsilon)$-approximation ($\approx 1.71+\varepsilon$) for the special case of TTSP(1,2) in which the lifetime is restricted to n. The latter is obtained by approximating a temporal path packing instead of a matching. The second approach improves these to $1.7+\varepsilon$ for the general case and to $13/8+\varepsilon = 1.625+\varepsilon$ when the lifetime is n. In all cases, $\varepsilon > 0$ is a small constant (not necessarily the same in all cases) adopted from the factors of the approximation algorithms for independent set and set packing. We leave as an interesting open problem whether a (3/2)-factor either for the general TTSP(1,2) or for the special case with lifetime restricted to n is within reach.

Apart from TTSP(1,2) we also consider the Temporal (Node) Exploration (abbreviated TEXP) problem, in which we are given a temporal graph (unweighted and with not necessarily complete instances) and the goal is to visit all nodes of the temporal graph, by possibly revisiting nodes, minimizing the arrival time (in the static case, appears as Graphic TSP in the literature). Though, in the static case, the decision version of the problem, asking whether a given graph is explorable, can be solved in linear time, we show that in the temporal case it becomes **NP**-complete. Additionally, in the static case, there is a $(3/2-\varepsilon)$-approximation for undirected graphs [GSS11] (and a more recent, 1.4-approximation [SV14], the best to date) and a $O(\log n/\log\log n)$ for directed [AGM+10]. In contrast to these approximations, we prove that there exists some constant $c > 0$ such that a special case of TEXP cannot be approximated within cn unless $\mathbf{P} = \mathbf{NP}$. Additionally, we prove that even the special case in which every instance of the temporal graph is strongly connected, cannot be approximated within $(2-\varepsilon)$, for every constant $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$. On the positive side, we show that TEXP can be approximated within the *dynamic diameter* of the temporal graph (see Section 2 for a formal definition of the dynamic diameter).

Finally, in the way to approaching the above two main problems, we also obtain several interesting side-results. In particular, we obtain a $[3/(5+\varepsilon)]$-approximation for temporal matching with time differences at least two between its edges, a $[1/(7/2+\varepsilon)]$-approximation for a special case of Temporal Path Packing (TPP) when the lifetime is restricted to n, and a (1/5)-approximation for Max-TTSP. Finally, we obtain two more inapproximability results. One states that there is some $\varepsilon > 0$, such that kTPP (a variation of TPP defined in Section 2.1) cannot be approximated within k^{ε}, for all $k \geq 3$, and the other that, for any polynomial time computable function $\alpha(n)$, Temporal Cycle Cover cannot be approximated within $\alpha(n)$, unless $\mathbf{P} = \mathbf{NP}$. To the best of our knowledge, all the aforementioned temporal problems are first studied in this work (i.e. all problems defined in Section 2.1).

1.2. Further Related Work

TSP in Static Graphs. The Traveling Salesman Problem is one of the most intensively studied and well-known combinatorial optimization problems. Although studied for more than sixty years, it continues to pose grant challenges. For the Symmetric TSP, where the graph is undirected and costs obey the triangle inequality, the best known approximation algorithm is still, after almost forty years of intensive effort to improve it, the celebrated 3/2 of Christofides [Chr76]. Only recently, Gharan *et al.* [GSS11] proved that the Graphic TSP special case in which the costs correspond to shortest path distances of some given graph can be approximated within $3/2 - \varepsilon$, for a small constant $\varepsilon > 0$ (which was further improved by subsequent works, e.g. [SV14]). For the Asymmetric TSP, the $O(\log n)$-approximation of Frieze *et al.* [FGM82] was the best known for almost three decades and was improved only recently to $O(\log n/\log\log n)$ by Asadpour *et al.* [AGM$^+$10]. In the special case with costs one and two that we consider in this work, the best known approximations are 8/7 for the symmetric case [BK06] and 5/4 for the asymmetric case [Blä04]. Both versions of the problem have been shown to be **APX**-hard [PY93]. It is also known that the TSP, ATSP, TSP(1,2) and ATSP(1,2) cannot be approximated within any factor less than 123/122, 75/74 [KLS13], 535/534, and 207/206 [KS12, KS13], respectively.

Labeled Graphs. Labeled graphs have been widely used in Computer Science and Mathematics, e.g. in Graph Coloring [MR02]. In our work, labels correspond to moments in time and the properties of labeled graphs that we consider are naturally *temporal properties*. Note, however, that any property of a graph labeled from a discrete set of labels corresponds to some temporal property if interpreted appropriately (even a proper coloring). Though we focus on properties with natural temporal meaning, our definitions are generic and do not exclude other, yet to be defined, properties that may prove important in the future. Several papers have considered labeled versions of polynomial-time solvable problems, in which the goal is to minimize/maximize the number of labels used by a solution. For example, the first labeled problem introduced in the literature was the Labeled Minimum Spanning Tree problem, which has several applications in communication network design. This problem is **NP**-hard and many complexity and approximability results have been proposed (see e.g. [BL97, KW98]). On the other hand, the Labeled Maximum Spanning Tree problem has been shown polynomial in [BL97]. In [BLWZ05], the authors proved that the Labeled Minimum Path problem is **NP**-hard and provided some exact and approximation algorithms. In [Mon05], it was proved that the Labeled Perfect Matching problem in bipartite graphs is **APX**-complete (see also [TIR78] for a related problem).

Continuous Availabilities (Intervals). Some authors have naturally assumed that an edge may be available for continuous time-intervals. The techniques used there are quite different than those needed in the discrete case [XFJ03, FT98].

Dynamic Distributed Networks. In recent years, there is a growing interest in distributed computing systems that are inherently dynamic. This has been mainly driven by the advent of low-cost wireless communication devices and the development of efficient wireless communication protocols. Apart from the huge amount of work that has been devoted to applications, there is also a steadily growing concrete set of theoretical work. In fact, the study of temporal graphs has been highly motivated by recent advances in the area of dynamic distributed networks. A notable set of recent works has studied (distributed) computation in *worst-case* dynamic networks in which the topology may change arbitrarily from round to round (see e.g. [KLO10, MCS14]). Population protocols [AAD$^+$06] and variants [MCS11a] are collections of passively mobile finite-state agents that compute something useful in the limit. In [MS14a], the authors studied the fundamental problem of *stable network construction* by a dynamic distributed computing system. They gave protocols (optimal in some cases) and lower bounds for several basic network construction problems such as *spanning line*, *spanning ring*, *spanning star*, and *regular network* and they proved several *universality* results by presenting generic protocols that are capable of simulating a Turing Machine (TM) and exploiting it in order to construct a large class of networks. Another interesting direction assumes random network dynamicity and the interest is on determining "good" properties of the dynamic network that hold with high probability and on designing protocols for distributed tasks [CMM$^+$08, AKL08]. For introductory texts the reader is referred to [CFQS12, MCS11b, Sch02, HS12].

Canadian Traveler Problem (CTP). The exploration problem studied here has some similarities to the Canadian Traveler Problem [PY91], as both consider the problem of exploring a graph with some dynamic characteristics. However, there is a great difference: in the CTP problem the algorithm is an online algorithm that does not know in advance the edge unavailabilities while in our case, we study offline algorithms knowing the entire temporal graph in advance. As a consequence, the performance of an algorithm for CTP is naturally given in terms of a competitive ratio, where the performance of the algorithm is compared to the performance of an optimal full-knowledge offline algorithm.

1.3. Organization

In Section 2, we formally define the model of temporal graphs under consideration and provide all further necessary definitions. In particular, in Section 2.1 we provide formal definitions of all temporal problems that we introduce and consider in this work. In Section 3, we consider the Temporal Exploration problem. We discuss that in the static case the decision version of the problem can be solved in linear time both in undirected and directed graphs and the optimization version is approximable in undirected graphs within $3/2-\varepsilon$ and in directed graphs within $O(\log n/\log\log n)$. In contrast to these, we prove that in temporal graphs the decision version of the problem becomes **NP**-complete (Section 3.1) while for the optimization version we provide hardness of approximation results. In particular, we prove in Section 3.3 that, in a special case of the problem, there exists some constant $c>0$ such that the optimum cannot be approximated within cn and that, even if we restrict attention to temporal graphs with strongly connected instances, there is no $(2-\varepsilon)$-factor approximation algorithm for Temporal Exploration, for every constant $\varepsilon>0$, both unless $\mathbf{P}=\mathbf{NP}$. On the positive side, we present in Section 3.4 a d-approximation algorithm, where d is the dynamic diameter of the temporal graph (see Section 2 for a definition of the dynamic diameter). Then, in Section 4 we introduce and study the Temporal Traveling Salesman Problem with Costs One and Two (abbreviated TTSP(1,2)) in weighted temporal graphs, which is obviously **APX**-hard as a generalization of ATSP(1,2) (**APX**-hardness holds also for the special case with lifetime restricted to n). The main approach that we follow to approximate TTSP(1,2) is to compute a temporal matching containing as many ones as possible and then patch the edges of the matching in a time-respecting way to obtain a TTSP tour. Compared to the trivial 3/2 approximation that one obtains by this approach in static graphs, due to the fact that Maximum Matching is in **P**, this is not the case in temporal graphs. In particular, in Section 4.1 we prove by reduction from Balanced 3SAT that computing such temporal matchings is **NP**-complete. Naturally, in the following sections, we look for approximations for temporal matchings. First, in Section 4.2 we give an approximation preserving reduction from the temporal matching problem to the problem of computing a maximum independent set in 5-claw free graphs. This gives us a $(\frac{1}{2+\varepsilon})$-approximation for matchings which then translates to a $(7/4+\varepsilon)$-approximation for TTSP(1,2). The same approach, in the special case in which the lifetime of the temporal graph is restricted to n, by expressing a path packing problem this time as an 8-claw free graph, gives us a $[1/(7/2+\varepsilon)]$-approximation for Temporal Path Packing with consecutive time-labels on its paths, which implies a $(12/7+\varepsilon)$-approximation for TTSP(1,2). Then, in Section 4.3 we follow an alternative route and reduce the temporal matching problem, in an approximation preserving way, to 4-Set Packing this time. This approach improves the previous factor for temporal matching to $\frac{3}{5+\varepsilon}$ and for TTSP(1,2) to $(1.7+\varepsilon)$ in the generic case and to $(13/8+\varepsilon)$ in the special case of lifetime restricted to n. Then, in Section 5 we give approximation algorithms and hardness results for other related temporal-graph problems. Finally, in Section 6 we conclude and give further research directions that are opened by our work.

2. Preliminaries

Definition 1. *A* temporal graph *(or* dynamic graph*) D is an ordered pair of disjoint sets (V,A) such that $A\subseteq\binom{V}{2}\times\mathbb{N}$. In case of a temporal digraph, instead of $\binom{V}{2}$ we use $V^2\backslash\{(u,u):u\in V\}$. The set V is the* *set of* vertices *(or* nodes*) and the set A is the set of* time-edges.

A temporal (di)graph $D = (V, A)$ can be also viewed as a static (underlying) graph $G_D = (V, E)$, where $E = \{e : (e, t) \in A$ for some $t \in \mathbb{N}\}$ contains all edges that appear at least once, together with a labeling $\lambda_D : E \to 2^{\mathbb{N}}$ defined as $\lambda_D(e) = \{t : (e, t) \in A\}$ (we omit the subscript D when no confusion can arise). We denote by $\lambda(E)$ the multiset of all labels assigned to the underlying graph by the labeling λ and by $|\lambda| = |\lambda(E)|$ their cardinality (i.e. $|\lambda| = \sum_{e \in E} |\lambda(e)|$). We also denote by $\lambda_{\min} = \min\{l \in \lambda(E)\}$ the minimum label and by $\lambda_{\max} = \max\{l \in \lambda(E)\}$ the maximum label of a temporal graph D. We define the *lifetime* (or *age*) of a temporal graph D as $\alpha(D) = \lambda_{\max} - \lambda_{\min} + 1$. Note that in case $\lambda_{\min} = 1$ we have $\alpha(D) = \lambda_{\max}$.

For every time $t \in \mathbb{N}$, we define the *t-th instance of a temporal graph $D = (V, A)$* as the static graph $D(t) = (V, A(t))$, where $A(t) = \{e : (e, t) \in A\}$ is the (possibly empty) set of all edges that appear in D at time t. A temporal graph $D = (V, A)$ may be also viewed as a *sequence of static graphs* $(G_1, G_2, \ldots, G_{\alpha(D)})$, where $G_i = D(\lambda_{\min} + i - 1)$ for all $1 \leq i \leq \alpha(D)$. Another, often convenient, representation of a temporal graph is the following.

Definition 2. *The* static expansion *of a temporal graph $D = (V, A)$ is a DAG $H = (S, E)$ defined as follows. If $V = \{u_1, u_2, \ldots, u_n\}$ then $S = \{u_{ij} : \lambda_{\min} - 1 \leq i \leq \lambda_{\max}, 1 \leq j \leq n\}$ and $E = \{(u_{(i-1)j}, u_{ij'}) : \lambda_{\min} \leq i \leq \lambda_{\max}$ and $j = j'$ or $(u_j, u_{j'}) \in A(i)\}$. From now on, we always ignore the vertical edges so in every static expansion we just assume that $E = \{(u_{(i-1)j}, u_{ij'}) : (u_j, u_{j'}) \in A(i)$ for some $\lambda_{\min} \leq i \leq \lambda_{\max}\}$ (see Figure 4 in Section 4.2.1 for an example).*

A *temporal* (or *time-respecting*) *walk* W of a temporal graph $D = (V, A)$ is an alternating sequence of nodes and times $(u_1, t_1, u_2, t_2, \ldots, u_{k-1}, t_{k-1}, u_k)$ where $(u_i u_{i+1}, t_i) \in A$, for all $1 \leq i \leq k - 1$, and $t_i < t_{i+1}$, for all $1 \leq i \leq k - 2$. We call $t_{k-1} - t_1 + 1$ the *duration* (or *temporal length*) of the walk W, t_1 its *departure time* and t_{k-1} its *arrival time*. A *journey* (or *temporal/time-respecting path*) J is a temporal walk with pairwise distinct nodes. In words, a journey of D is a path of G_D that uses strictly increasing edge-labels. A (u, v)-journey J is called *foremost from time $t \in \mathbb{N}$* if it departs after time t and its arrival time is minimized. The *temporal distance* from a node u at time t (from now on, this will be explicitly referred to as the *time-node* (u, t)) to a node v is defined as the duration of a foremost (u, v)-journey from time t. We say that a temporal graph $D = (V, A)$ has *dynamic diameter* d, if d is the minimum integer for which it holds that the temporal distance from every time-node $(u, t) \in V \times \{0, 1, \ldots, \alpha(D) - d\}$ to every node $v \in V$ is at most d. A *temporal matching* of a temporal graph $D = (V, A)$ is a set of time-edges $M = \{(e_1, t_1), (e_2, t_2), \ldots, (e_k, t_k)\}$, such that $(e_i, t_i) \in A$, for all $1 \leq i \leq k$, $t_i \neq t_j$, for all $1 \leq i < j \leq k$, and $\{e_1, e_2, \ldots, e_k\}$ is a matching of G_D.

Similarly to weighted graphs we may define *weighted temporal graphs* by introducing a (temporal) cost function $c : A \to \mathcal{C}$, where $\mathcal{C}$ denotes the range of the costs, e.g. $\mathcal{C} = \mathbb{N}$. Note that an equivalent representation of a temporal graph $D = (V, A)$ is to assume that $D(t)$ is complete for all $1 \leq t \leq \alpha(D)$ and assign cost 1 to all $a \in A$ and cost ∞ to all missing time-edges. A temporal graph $D = (V, A)$ is called *complete* if $D(t)$ is complete for all $1 \leq t \leq \alpha(D)$. Moreover, a temporal graph D is called *continuously connected* (aka *1-interval connected* [KLO10, MCS14]) if $D(t)$ is connected for all $1 \leq t \leq \alpha(D)$. In these cases, we may also say that *D has complete/connected instances*.

Throughout the text, unless otherwise stated, we denote by n the number of nodes of (temporal) (di)graphs. When no confusion may arise, we use the term *edge* for both undirected edges and arcs.

Given an instance I of an optimization problem Π, we denote by $\mathrm{OPT}_{\Pi}(I)$ the objective function value of an optimal solution to instance I and by $\mathrm{ALG}_{\Pi}(I)$ the objective function value of the solution produced by an algorithm under consideration. From now on, we will shorten this to OPT_{Π} or just OPT (similarly for ALG). Let $\delta \geq 1$ ($\delta \leq 1$) be a positive real number. A polynomial-time algorithm $\mathcal{A}$ is said to be a *δ-factor approximation algorithm* for a minimization (maximization) problem Π, if on each instance I, $\mathcal{A}$ produces a feasible solution s for I such that $\mathrm{ALG} \leq \delta \cdot \mathrm{OPT}$ ($\mathrm{ALG} \geq \delta \cdot \mathrm{OPT}$). We may also call $\mathcal{A}$ a *δ-approximation algorithm* for Π or say that *$\mathcal{A}$ approximates Π within δ*.

2.1. Problem Definitions

Our main focus in this work is on the following two problems, which are temporal variants of well-known combinatorial optimization problems.

Temporal Exploration - TEXP. Given a temporal graph $D = (V, A)$ (directed or undirected) and a source node $s \in V$, find a temporal walk that begins from s and visits all nodes minimizing the arrival time. So, throughout this work, by "exploring a temporal graph D" we always mean "exploring $V[D]$". We also study an interesting special case of TEXP in which the temporal graph is continuously connected.

TTSP(1,2). Given a complete temporal graph $D = (V, A)$ and a cost function $c : A \to \{1, 2\}$ find a temporal TSP tour of minimum total cost. A TSP tour $(u_1, t_1, u_2, t_2, \ldots, t_{n-1}, u_n, t_n, u_1)$ is temporal if $t_i < t_{i+1}$ for all $1 \leq i \leq n-1$ (clearly, it must hold that $\alpha(D) \geq n-1$). The cost of such a TSP tour is $\sum_{1 \leq i \leq n} c((u_i, u_{i+1}), t_i)$, where $u_{n+1} = u_1$. In general, the lifetime of D is not restricted, however in this work we also study the special case of TTSP(1,2) in which the lifetime is limited to n.

The following are intermediate temporal problems that we introduce and study in the way to approaching the above main problems.

Temporal Matching - TEM. Given a temporal graph $D = (V, A)$ decide whether there is a maximum matching M of G_D that can be made temporal by selecting a single label $l \in \lambda(e)$ for every edge $e \in M$.

Max-TEM. Given a temporal graph $D = (V, A)$ find a temporal matching of maximum cardinality.

Max-TEM($\geq k$). As Max-TEM with the only difference being that a feasible solution is now any temporal matching $M = \{(e_1, t_1), (e_2, t_2), \ldots, (e_h, t_h)\}$ for which it holds that there is a permutation $t_{i_1}, t_{i_2}, \ldots, t_{i_h}$ of the t_js such that $t_{i_{(l+1)}} \geq t_{i_l} + k$ for all $1 \leq l \leq h-1$. That is, any two consecutive edges in the time-respecting ordering of the matching should have a time difference of at least k. Observe that Max-TEM(≥ 1) is equivalent to Max-TEM.

Temporal Path Packing - TPP. We are given a temporal graph and we want to find time and node disjoint time-respecting paths maximizing the number of edges used. By time disjoint we require that they correspond to distinct intervals that differ by ≥ 2 in time.

Time-Consecutive Temporal Path Packing - CTPP. AS in TPP with the additional constraint that any path in the solution can only use *consecutive* time-labels.

k-Temporal Path Packing - kTPP. Given a temporal graph find a maximum-cardinality set of time-respecting node-disjoint and time-disjoint paths (pairwise) each of length at least k. (we have a family of problems one for each $k \leq n-1$)

Minimum Temporal Cycle Cover - TCC. Given a complete temporal graph and a cost function (general nonnegative) $c : A \to \mathbb{N}$ compute a minimum cost temporal cycle cover $c_1, c_2, \ldots, c_k$. We require such a cycle cover to satisfy (i) for all $u \in V$, u is covered by some c_i, (ii) c_i is a time-respecting cycle for all i, and (iii) $maxl(c_i) < minl(c_{i+1})$ for all i, where $minl(c)$ ($maxl(c)$) is the minimum (maximum, resp.) label in c.

Max-TTSP. The same as TTSP(1,2) however the weights are now *general nonnegative integers* and we want a TTSP tour of *maximum weight.*

3. Exploration of Temporal Graphs

In this section, we study the Temporal Exploration (TEXP) problem, in which we are given a temporal graph (unweighted) and the goal is to visit all nodes of the temporal graph by possibly revisiting nodes minimizing the arrival time. In contrast to several positive results known for the static case, which we discuss, we show that in temporal graphs the problem is quite hard. In particular, we show that the decision version of the problem **NP**-complete and we give two hardness of approximation results for the optimization version, one for the generic case and another for the special case in which the temporal graph is continuously connected. On the positive side, we show how to approximate the optimum of the generic instances within the dynamic diameter of the temporal graph.

3.1. Deciding Explorability is Hard in Temporal Graphs

Note that a walk in the (Temporal) Exploration is allowed to revisit nodes several times. Let us first focus on static graphs. Consider the decision version DEXP of Exploration in which the goal is to decide whether a given graph is explorable.

Proposition 1. *DEXP and finding an arbitrary solution can be solved in linear time for both undirected and directed static graphs.*

Proof. In both undirected and directed graphs the given graph G is explorable iff there is some walk from the source s visiting all nodes. In the undirected case this is equivalent to determining connectivity of G. Moreover, to find a solution, it suffices to compute a spanning tree of G and then perform searching on the tree from s, e.g. DFS. In the directed case, compute the DAG G' of strongly connected components of G. Then take a topological sort of G'. There is a directed walk from s to all nodes of G iff there is in G' a hamiltonian path from the component of s, i.e. iff there is a unique topological sort $C_1, C_2, \ldots, C_h$, where $s \in C_1$ and $(C_i, C_{i+1}) \in E[G']$ (i.e. directed edge) for all $1 \leq i \leq h-1$. All these can be computed in linear time. □

Next consider the analogous problem DTEXP of deciding whether exploring the nodes of a temporal graph $D = (V, A)$ is feasible. We are given again a source node s. Interestingly, this linear-time problem for static graphs becomes **NP**-hard in temporal graphs.

Proposition 2. *DTEXP is* **NP***-complete.*

Proof. Clearly in **NP** by using a temporal walk that begins from s and visits all nodes as the certificate. Its length is $O(n^2)$ as it is the concatenation of $n-1$ journeys each one of them beginning from a node that has just been visited for the first time and leading to an unvisited node, so any such journey in the worst case revisits all visited nodes before arriving at an unvisited one.

For the hardness part we reduce Hamiltonian Path (abbreviated Hampath) to DTEXP. Given a graph G and a source node s construct a temporal graph D by taking $n-1$ copies of G, that is $D(t) = G$ for all $1 \leq t \leq n-1$. A crucial observation is that the lifetime of D is restricted to $n-1$.

($\Rightarrow$) If G has a hamiltonian path from s, then G' has clearly a temporal hamiltonian path from the time-node $(s, 0)$ visiting one node at a time and covering all nodes in $n-1$ steps.

($\Leftarrow$) If D has a temporal walk W that visits all nodes, then W cannot repeat nodes because the lifetime of D does not suffice to fulfill at the same time covering and repetitions. To this end, observe that W has to visit $n-1$ distinct nodes in $n-1$ steps and if two or more steps occupy the same node then necessarily some other node must be left without a step, i.e. unvisited. We conclude that any such walk must correspond to a hamiltonian path of G. □

In principle, in an unrestricted temporal graph, connectivity may be modified or even totally abolished at any time. Exploring the temporal graph that we constructed in Proposition 2 is equivalent to requiring an exploration of a static graph to succeed within $n-1$ steps (a restriction that would make the problem **NP**-hard even in the static case). Moreover, observe that whenever G is connected the resulting D has connected instances, thus we obtain as a corollary that small lifetime makes exploration hard also in continuously connected temporal graphs.

3.2. Approximate Exploration in Static Graphs

On the other hand, for undirected static graphs, there is a trivial 2-factor approximation algorithm on the shortest exploration walk even with weights. The idea is to compute an MST T. Then to explore V via T we have to pay at most $2c(T) - 1$. Then, if OPT denotes the cost of the minimum exploration and ALG the cost of the exploration produced by the algorithm, we have that $c(T) \leq \text{OPT} \Rightarrow \text{ALG} \leq 2\text{OPT} - 1$. To obtain better approximations for undirected graphs and any approximations for directed ones one has to think a bit more.

Consider a strongly connected static digraph $G = (V, E)$ that we want to explore from some $u_1 \in V$. An exploration of G is just a permutation of the nodes $u_1 = v_1, v_2, v_3, \ldots, v_n$ encoding the order in which we decide to visit the nodes (i.e. order of first visit).

Assume that we first decide on the order. Then what we actually pay is $\sum_{1 \leq i \leq n-1} d(v_i, v_{i+1})$ where $d(u, v)$ is the length of the shortest path from u to v. For example, if there were a hamiltonian path from u_1 then we would just have to decide on the order corresponding to that hamiltonian path and then just pay the pairwise shortest paths each of length 1.

Based on this idea, we construct a complete digraph $G' = (V', E')$ where every $(u, v) \in E'$ has a cost $c(u, v)$ equal to the shortest path from u to v in G. As G is strongly connected all edges are assigned a natural number between 1 and $n-1$. Moreover, G' respects the triangle inequality: $c(u, v) \leq c(u, w) + c(w, v)$ because a shortest path from u to v cannot be longer than a shortest path from u to w and then a shortest path from w to v.

Now denote by OPT_G the optimum exploration of G and by $\text{OPT}_{G'}$ the optimum directed TSP path from u_1 in G'.

1. $\text{OPT}_{G'} \leq \text{OPT}_G$ because the permutation corresponding to the optimum exploration in G is a hamiltonian path of G' and the latter cannot have greater cost because it uses shortest paths to move from one node to the other.

2. Every hamiltonian path of G' corresponds to an exploration of G of the same cost. Simply follow in G the nodes in the order of the hamiltonian path by always using shortest paths. Clearly, you may end up with a different ordering because you may choose some shortest paths that visit nodes in different order than the hamiltonian path however you do not increase the cost.

Algorithm: Construct G' from G as above. Find a hamiltonian path P of G' and output an exploration that corresponds to that path. Let ALG be the cost of the solution. Assume that for the returned path we have the guarantee that $c(P) \leq k\text{OPT}_{G'}$. Clearly, $\text{ALG} = c(P) \leq k\text{OPT}_{G'} \leq k\text{OPT}_G$ which implies that any k-factor approximation for ATSP with triangle inequality and costs between 1 and $n-1$ gives a k-factor approximation algorithm for exploration of strongly connected static digraphs.

As a corollary of this we get that the $\log n$-factor approximation of Frieze et al. [FGM82] gives immediately a $\log n$-factor for our exploration problem (the same for the recent $\log n / \log \log n$ algorithm [AGM$^+$10]). Note that ATSP algorithms return a hamiltonian cycle however we can drop the last edge (the one that returns back to u_1) and the resulting path can be more expensive than the optimum hamiltonian path by at most an additive $n-2$ (which does not affect much the above result). See also [GSS11] for an algorithm that gives a $< 3/2$ approximation for exploring undirected graphs.

Remark 1. *There are strongly connected digraphs in which the optimum exploration path has length $\Theta(n^2)$: consider a directed path $u_1, u_2, u_4, u_6, \ldots, u_{n/2}$ and give to every u_i on the path a leaf, i.e. $(u_2, u_3), (u_4, u_5), \ldots$ Then connect every such leaf back to u_1. Note that every time we visit a leaf, we have to return back to the origin and revisit edges, i.e. the first edge on the line has to be visited $n/2$ times, the second $n/2 - 1$, and so on, so we get that all exploration paths of such a graph have length $\Theta(n^2)$.*

This is quite different compared to undirected graphs where there is always an exploration path of length $\leq 2(n-1) - 1$ (there we just have to do DFS on any spanning tree).

The above immediately implies that there are also temporal graphs in which every exploration walk has duration $\Theta(n^2)$.

3.3. Hardness of Approximate Temporal Exploration

A simple modification of the reduction of Proposition 2 can establish that TEXP cannot be approximated within any polynomial factor unless $\mathbf{P} = \mathbf{NP}$. Intuitively, when the input graph is hamiltonian, the exploration requires only $n-1$ steps by following the hamiltonian path. On the other hand, when it is not hamiltonian, at step $n-1$ at least one node has not been visited yet, so, if beginning from step n the temporal graph becomes empty for a polynomial number of steps, e.g. n^k, and then complete for $n-1$ steps, then in the non-hamiltonian case the exploration will have to wait n^k steps until it can complete the exploration. In light of this strong negative result, we investigate some interesting special cases of TEXP, and show that the problem is still hard to approximate.

Theorem 1. *There exists some constant $c > 0$ such that TEXP cannot be approximated within cn, even in temporal graphs consisting, in every instance, of two strongly connected components (weakly connected with each other), unless* $\mathbf{P} = \mathbf{NP}$.

Proof. We present a gap introducing reduction from Hampath. We are given a strongly connected graph G and a source node s on which we want to decide whether there exists a hamiltonian path from s. We construct a temporal graph D consisting of a static part $G_1 = G$ (the left part in Figure 2) and a second dynamic part G_2 (the right part of Figure 2). Denote by n, n_1, and n_2 the orders of D, G_1, and G_2, respectively, and assume that $n_1 = n_2$ and that n_1 is odd (the even case is similar). So, it holds that $n = n_1 + n_2 = 2n_1 = 2n_2$. If G is hamiltonian then there is a path in $G_1 = G$ that arrives at v_1 in n_1 steps. Then by moving to v_2 and following the edges going down (observe that $n_2 = 2m+1$, where m is the number of nodes in each of the two parallel columns of G_2), i.e. $v_1, v_2, v_3, \ldots, v_m, v_{m+1}, v_{m+2}$, we may arrive at the bottom right corner v_{m+2} at step $n_1 + m + 1 = (5n_1+1)/2 = (5n+2)/4$ (which is integer whenever n is two times an odd number). Now beginning from v_{m+2} let every edge going up the right column, (v_{m+i+1}, v_{m+i+2}) for $1 \le i \le n_2 - m - 2$, appear only at times $[1, (5n+2)/4 + i]$, that is the first edge appears only up to time $(5n+2)/4 + 1$ (inclusive), the second only up to time $(5n+2)/4 + 2$, and so on, the last appearing up to time $n_1 + m + 1 + (m-1) = n - 1$.

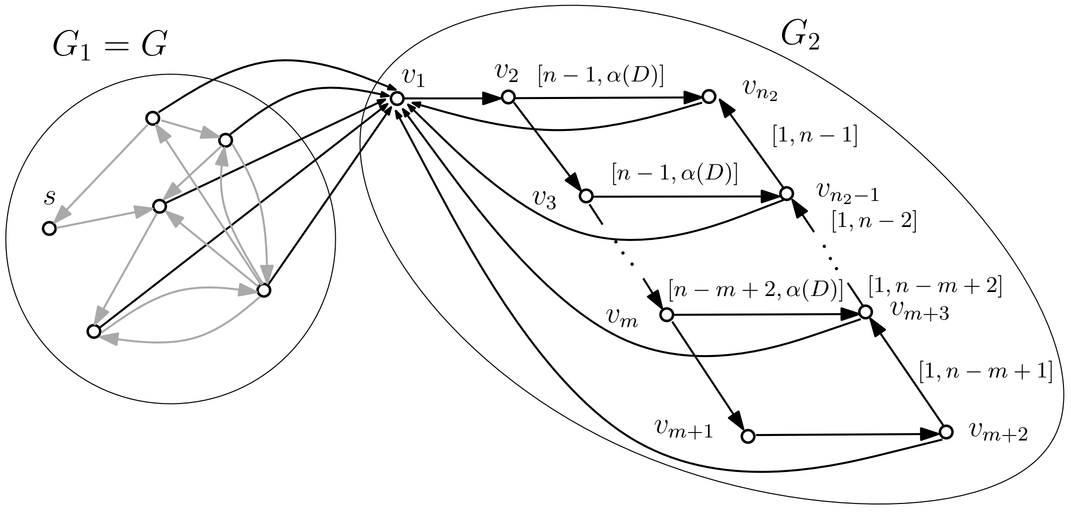

Figure 2: The temporal graph $D = (V, A)$ constructed by the reduction of Theorem 1. G_1, on the left part of D, is the instance of Hampath. The right part, G_2, together with the edges leading from G_1 to G_2 is the actual construction of the reduction. G_2 has an entry node v_1 and two vertical columns each consisting of $m = (n_2 - 1)/2$ nodes. All solid edges of G_2 are static (i.e. appear from time 1 up to the lifetime $\alpha(D)$). On the other hand, each of the horizontal edges appears at some time and remains static from that point on (these edges were erroneously reported static in [MS14b]). Also, the edges going up the right column are dynamic. Each of these appears from the beginning up to a time moment indicated by its interval-label. Observe that in the presence of these edges, V_2 can be explored from v_1 in $n_2 - 1$ steps while in their absence any exploration takes time $\Theta(n^2)$.

So, we have:

($\Rightarrow$) If G is hamiltonian then we can arrive at the first node of G_2 (i.e. v_1) as fast as possible and then we can again travel as fast as possible down the "left" nodes of G_2 to catch up the "right" edges going up. In this case we explore optimally in $n-1$ steps, i.e. $\mathrm{OPT} = n-1$.

($\Leftarrow$) On the other hand, if G is not hamiltonian then there is not enough time to catch up the right edges going up and the optimum exploration of G_2 takes now time $\Theta(n^2)$. To see that there is not enough time to catch up those edges, observe that each of the horizontal edges of G_2 first appears at the time when its subsequent edge going up will be unavailable in the next step. So, these edges cannot be used to catch up the edges going up. The only edge that can be used to catch up the vertical edges is the static vertical edge at the bottom, but the walk cannot arrive there before time $n-m+1$, that is when it arrives it is too late to catch up the path going up. To see that the optimum exploration of G_2 takes now time $\Theta(n^2)$, observe that in the absence of the rightmost vertical edges, whenever we visit a right node we have to return to v_1 and revisit part of the left nodes. In particular, every edge of the left part is revisited a number of times equal to the number of right nodes that lie below it. The reason that we arrive late at G_2 is that in order to move from G_1 to G_2 all nodes in G_1 must have been explored first because there are no inverse edges leading back from G_2 to G_1. Thus, there exists some constant $c > 0$ for which (for sufficiently large n) $\mathrm{OPT} \geq cn^2 > (cn)(n-1)$.

Moreover, G_2 is strongly connected in every instance, as required. To see this, observe that up to time $n-m+1$, all right edges are still available, so the hamiltonian cycle (in G_2) $v_1, v_2, v_3, \ldots, v_{m+1}, v_{m+2}, v_{m+3}, \ldots, v_{n_2}, v_1$ is available, therefore up to time $n-m+1$, G_2 is strongly connected. From time $n-m+2$, in every step, the lowest available right edge becomes unavailable and the lowest unavailable horizontal edge (incident to the head of the eliminated vertical) becomes available. These instances are still strongly connected, because the prefix of the right path that has become unavailable can be replaced by the corresponding (v_{m+1+j}, v_1) backward edges and every time re-traversing the left path going down.

As we have introduced a gap of factor cn, it follows that a (cn)-approximation for TEXP could be used to decide Hampath which cannot be the case unless $\mathbf{P} = \mathbf{NP}$. $\square$

Observe that in the above reduction we were free to break at some point the strong connectivity of the temporal graph. In particular, there is never a path from G_2 to G_1, and we did this in order to prevent the walk from deferring the exploration of some nodes of G_1 for the end and ensure in this way that, in the non-hamiltonian case, the walk always arrives late at G_2. If one restricts attention to the special case of TEXP in which the temporal graph is continuously (strongly) connected, the above freedom is lost, because now there should always be a way to return back to G_1. So, to prove an inapproximability result for this special case we have to develop a different reduction technique.

Theorem 2. *For every constant $\varepsilon > 0$, there is no $(2-\varepsilon)$-factor approximation algorithm for TEXP in continuously (strongly) connected temporal graphs unless $\mathbf{P} = \mathbf{NP}$.*

Proof. The reduction is again from Hampath. We prove that a $(2-\varepsilon)$-factor approximation for TEXP in continuously connected temporal graphs could be used to decide Hampath. Let (G, s) be an instance of Hampath. We construct an instance of TEXP consisting of a continuously strongly connected temporal graph $D = (V, A)$ and a source node s'. D consists of three static graphs T_1, T_2, and T_3 as illustrated in Figure 3. The first graph T_1 (Figure 3(a)) consists of $G_1 = G$ and a set V_2 of additional nodes, i.e. $V = V_1 \cup V_2$. Denote by n, n_1, and n_2 the cardinalities of V, V_1, and V_2, respectively. For the time being it suffices to assume that $n_2 > n_1$. We set $s' = s$. We connect every node of V_1 to the leftmost node of V_2, then continue with a directed path spanning V_2 (i.e. a hamiltonian path on V_2), and finally we connect the rightmost node of V_2 to each node of V_1. T_1 persists until time $n_1 - 1$, that is $D(t) = T_1$ for all $1 \leq t \leq n_1 - 1$. Then, at time n_1, D changes to the second graph T_2 (Figure 3(b)) which is the same as T_1 without the internal edges of set V_1 (those are the edges of G that were present in T_1). T_2 persists until time $n_2 - 1$, that is $D(t) = T_2$ for all $n_1 \leq t \leq n_2 - 1$. Finally, at time n_2, D changes to the third graph T_3 (Figure 3(c)) in which each of V_1 and V_2 has its nodes connected by a line of 2-cycles and the left endpoints of the two sets are also connected by a 2-cycle. T_3 is preserved up to the lifetime of D, that is $D(t) = T_3$

for all $n_2 \leq t \leq \alpha(D)$. To ensure explorability of D, it suffices to set $\alpha(D) = 2n_2 + n_1$. Note that D is a continuously strongly connected temporal graph because T_1, T_2, and T_3 are strongly connected graphs.

($\Rightarrow$) If G is hamiltonian, then the hamiltonian path of G_1, followed by an edge leading from V_1 to V_2, and finally followed by the hamiltonian path on V_2 gives a hamiltonian journey of D and thus V can be explored optimally in $n_1 + n_2 - 1$ steps.

($\Leftarrow$) If G is not hamiltonian, then we prove that in this case the optimum exploration needs at least $2n_2 + 1$ steps. Observe that by time $n_1 - 1$ the exploration cannot have visited all nodes of V_1 because G_1 is not hamiltonian from s (Figure 3(a)). This remains true until time $n_2 - 1$, because in the interval $[n_1, n_2 - 1]$ the only edges that lead to nodes in V_1 cannot have been reached before time n_2 (Figure 3(b)). So, by time $n_2 - 1$ there is an unvisited node in V_1. Moreover, by the same time the rightmost node of V_2 is also unvisited because the temporal distance from $(s, 0)$ to it is n_2. Then, even if at time n_2 the exploration hits one of them, the other is at distance $\geq n_2 + 1$ because the leftmost node of V_1 in Figure 3(c) is s. So, in total, at least $2n_2 + 1$ steps are needed to visit all nodes of D.

It remains to prove that the above reduction can be adjusted to introduce the claimed gap. As ε is a constant, we can restrict attention to instances of HAMPATH of order at least $2/\varepsilon$ and provide a gap introducing reduction from those instances (which obviously still remain hard to decide), that is $n_1 \geq 2/\varepsilon \Rightarrow \varepsilon \geq 2/n_1 \Rightarrow 2 - \varepsilon \leq 2 - (2/n_1)$. Moreover, in the above reduction set $n_2 = n_1^2 + n_1$ (observe that we can set n_2 equal to any polynomial-time computable function of n_1). So, by what has been proved so far, we have that:

- If G is hamiltonian, then $\text{OPT} = n_1 + n_2 - 1 = n_1^2 + 2n_1 - 1$.
- If G is not hamiltonian, then $\text{OPT} \geq 2n_2 + 1 = 2(n_1^2 + n_1) + 1 > 2(n_1^2 + n_1)$.

Consider the hamiltonian case. As $2 - \varepsilon \leq 2 - (2/n_1)$ we have

$$\begin{aligned}(2-\varepsilon)(n_1^2 + 2n_1 - 1) &\leq (2 - \frac{2}{n_1})(n_1^2 + 2n_1 - 1) \\ &= 2n_1^2 + 4n_1 - 2 - 2n_1 - 4 + \frac{2}{n_1} \\ &= 2(n_1^2 + n_1) + (\frac{2}{n_1} - 6) \\ &\leq 2(n_1^2 + n_1).\end{aligned}$$

Thus, whenever G is hamiltonian, the $(2-\varepsilon)$-approximation algorithm returns a solution of cost $\leq (2-\varepsilon)\text{OPT} = (2-\varepsilon)(n_1^2 + 2n_1 - 1) \leq 2(n_1^2 + n_1)$. On the other hand, whenever G is not hamiltonian, $\text{OPT} > 2(n_1^2 + n_1)$ and thus also the solution returned by the algorithm must have cost $> 2(n_1^2 + n_1)$. We conclude that if we had a $(2-\varepsilon)$-approximation algorithm we could decide the instances of HAMPATH of order at least $2/\varepsilon$ in polynomial time by comparing the objective of the algorithm's solution to the polynomial-time computable $2(n_1^2 + n_1)$ threshold. This cannot be the case unless $\mathbf{P} = \mathbf{NP}$. □

Remark 2. *It seems that we cannot hope for a stronger (than 2) hardness of approximation for continuously connected temporal graphs by this particular reduction technique for the following reason. A non-hamiltonian G_1 may differ from a hamiltonian one for only delay 1 to visit all nodes of G_1. However, in the non-hamiltonian case, the exploration may choose to visit only $n - 2$ nodes in precisely the same steps as in the hamiltonian case and then follow the optimal path that is also followed in the hamiltonian case to visit all remaining nodes (nodes in V_2) optimally. So in the non-hamiltonian case we can do precisely what we can do in the hamiltonian one if we decide to leave a V_1 node unvisited for sufficient time. Then when we visit all V_2 nodes, continuous connectivity guarantees that we can visit the unique remaining node in V_1 in at most $n_1 + n_2 - 1$ steps. So, given that we visited all other nodes also in $n_1 + n_2 - 1$ steps (optimally), in another $n_1 + n_2 - 1$ we succeed. Thus it seems that in continuously connected (due to dynamic diameter $n_1 + n_2 - 1$) we can succeed in some non-hamiltonian instances without producing more than a 2-gap. On the other hand, a reduction that looks more into the details of G_1 (or that reduces from some other problem) could possibly give a stronger inapproximability.*

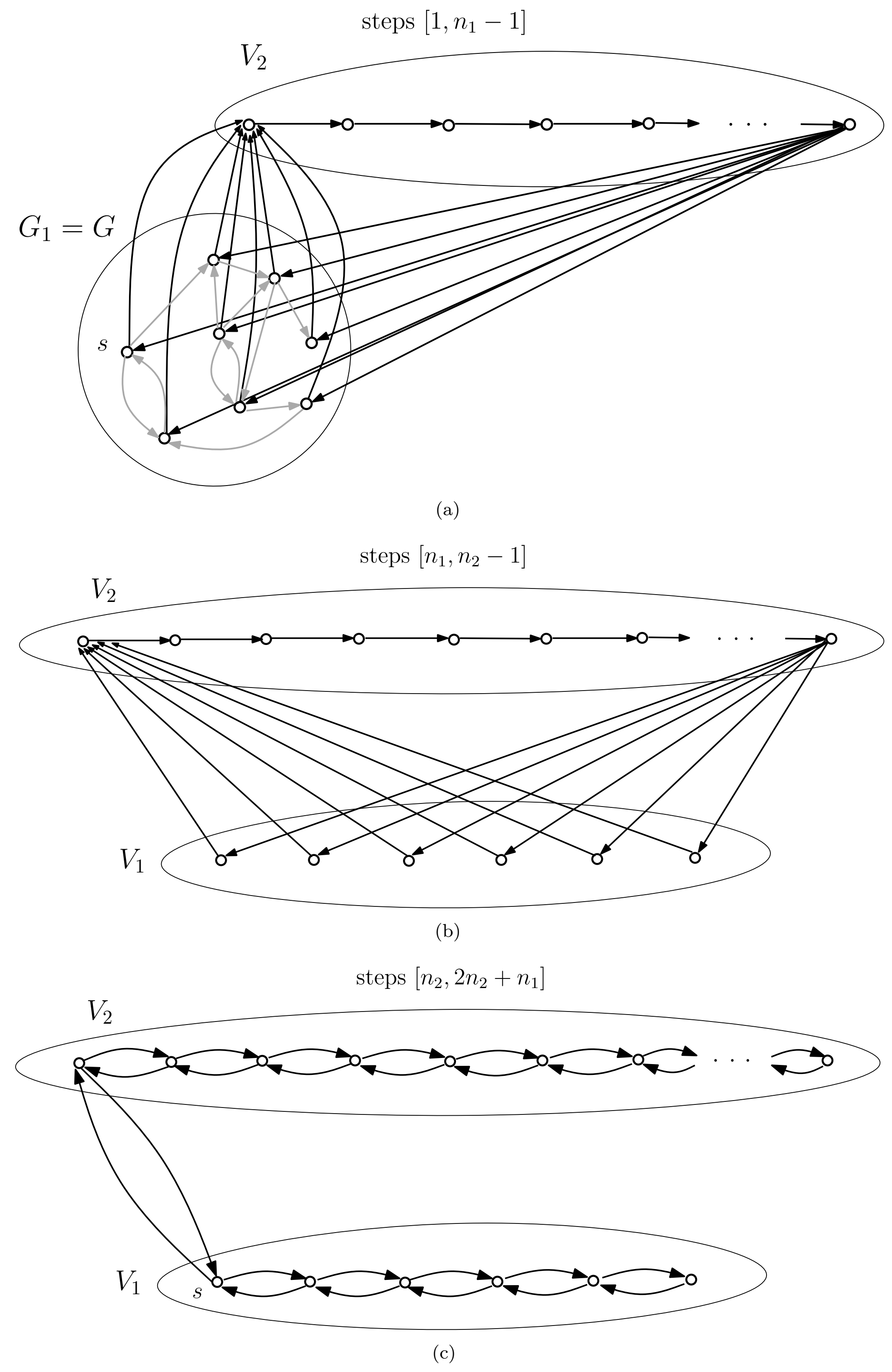

Figure 3: The temporal graph constructed by the reduction of Theorem 2. (a) T_1 (b) T_2 (c) T_3

3.4. A d-factor Approximation Algorithm for TEXP

Observe now that if a temporal graph is continuously (strongly) connected then it is explorable provided its lifetime is sufficient. In general, every temporal graph with dynamic diameter upper bounded by d and sufficient lifetime is explorable. In fact, there exists an exploration walk that succeeds in at most $(n-1)d$ steps. For continuously connected temporal graphs the bound is $(n-1)^2$ because $d \leq n-1$.

We now present an algorithm (see Algorithm 1) that given a temporal graph with dynamic diameter d and a source node s, outputs such a temporal exploration walk from s. The idea is as follows. Consider the set S containing the nodes visited so far, i.e. initially $S = \{s\}$. Take any $u \in V \backslash S$. As the dynamic diameter is d, there must be a journey from $(s, 0)$ to u arriving at most by time d. Follow such a journey, by computing a foremost journey from s to u from time 0, and put any visited node in S. Now the exploration walk is on u. Take any $v \in V \backslash S$ and do the same as before beginning now from u. Repeat until $S = V$. Observe that in at most $(n-1)d$ steps all nodes must have been visited (though the actual time-bound may be better).

Algorithm 1 d-approximation for TEXP

Require: Temporal graph $D = (V, A)$ with dynamic diameter upper bounded by d and $\alpha(D) \geq (n-1)d$, source node $s \in V$

Ensure: A temporal exploration walk from s

```
1: $S \leftarrow \{s\}$, $x \leftarrow s$, $t \leftarrow 0$, $W = \emptyset$
2: while $S \neq V$ do
3:     Pick any $u \in V \backslash S$
4:     Find a foremost journey from $(x, t)$ to $u$. Let the journey be $J$ and its arrival time $a$. // one must
       arrive at most by time $t + d$
5:     For all $v \in J \backslash S$ do $S \leftarrow S \cup \{v\}$
6:     $t \leftarrow a$
7:     $W \leftarrow W \circ J$
8:     $x \leftarrow u$
9: end while
10: Output $W$
```

Theorem 3. *Algorithm 1 is a d-factor approximation algorithm for TEXP restricted to temporal graphs with dynamic diameter upper bounded by* d *and lifetime at least* $(n-1)d$.

Proof. Clearly, in every execution of the while loop an unvisited node is visited in at most d steps. Since there are $n-1$ nodes to be visited the algorithm outputs an exploration walk W of duration at most $(n-1)d$. Moreover, any exploration walk including the optimum needs time at least $n-1$ to visit all nodes. Finally, the while loop is executed $n-1$ times thus the algorithm terminates in time $(n-1)K$, where K is the time needed by the foremost journey algorithm (which has been proven polynomial in [MMCS13]). □

4. Temporal Traveling Salesman with Costs One and Two

In this section, we deal with TTSP(1,2) which is a generalization of the well known ATSP(1,2) to weighted temporal graphs and, as such, it is **APX**-hard. Recall that in TTSP(1,2) we are given a complete temporal graph $D = (V, A)$, with its time-edges weighted according to a cost function $c : A \rightarrow \{1, 2\}$, and we are asked to find a temporal TSP tour of minimum total cost.

4.1. Computing Temporal Matchings is Hard

Most of our approximation algorithms for TTSP(1,2) (presented in Sections 4.2 and 4.3) compute first a temporal matching that uses as many 1s as possible and then extend it into a TTSP tour by connecting the edges of the matching in a time-respecting way. In this section, we show that, in contrast to static matchings that can be computed in polynomial time, finding a maximum temporal matching is hard and, therefore, we can only hope for approximate solutions.

First consider the Union Labeled Matching problem (ULM): Given a graph $G = (V, E)$, labels $L = \{1, 2, ..., h\}$, and a labeling $\lambda : E \to 2^L$, decide whether there is a maximum matching M of G s.t. $\bigcup_{e \in M} \lambda(e) = L$, that is a maximum matching that covers all labels.

Lemma 1. *ULM is* **NP**-*complete.*

Proof. The reduction is from 3SAT. Given a 3CNF formula ϕ with m variables x_i and z clauses C_l, we construct a labeled bipartite graph $G = ((X, Y), E)$, where X has a node for each one of the m variables and Y has $2m$ nodes corresponding to the truth values $0, 1$ repeated m times. Let $u_i \in X$ denote the node of variable x_i and let v_{i0} and v_{i1} be the nodes from Y corresponding to it. Connect node u_i to v_{i0} and v_{i1}, that is $E = \bigcup_{1 \le i \le m} \{u_i v_{i0}, u_i v_{i1}\}$. Label edge $u_i v_{ij}$ by a subset of $L = \{1, 2, \dots, z\}$ corresponding to the indices of the clauses that variable x_i satisfies when its truth value is set to j.

($\Rightarrow$) If ϕ is satisfiable, then there is an assignment f of truth values to the variables that satisfies all clauses. Consider the matching $M = \{u_i v_{ij} : f(x_i) = j\}$. M is maximum because f assigns a truth value to every variable, so we get an edge for every $u_i \in X$. Moreover, $\bigcup_{e \in M} \lambda(e) = \bigcup_{1 \le i \le m} \{l : C_l$ is satisfied when x_i is set to $f(x_i)\} = L$, because ϕ is satisfied by f. We conclude that M is a maximum matching of G covering all labels.

($\Leftarrow$) If there is in G a maximum matching M covering all labels, then the truth assignment f defined as $f(x_i) = j$ iff $u_i v_{ij} \in M$ satisfies all clauses. $\square$

Now consider the Balanced ULM (BULM) problem in which we have a bipartite $G = ((X, Y), E)$, every node $u_i \in X$ has precisely two neighbors $v_{ij} \in Y$, and additionally both edges of u have the same number of labels.

Lemma 2. *BULM is* **NP**-*complete.*

Proof. The reduction is the same as in Lemma 1, but this time from the **NP**-complete Balanced 3SAT in which every variable x_i appears n_i times negated and n_i times non-negated. [1] $\square$

Now consider the Temporal Matching (TEM) problem:

Theorem 4. *TEM is* **NP**-*complete.*

Proof. Clearly in **NP**. For the hardness part, we reduce BULM to TEM. We have a bipartite graph $G = ((X, Y), E)$ labeled by $\lambda : E \to 2^L$ from a set of labels $L = \{1, 2, \dots, h\}$, every node $u_i \in X$ has precisely two neighbors $v_{i1}, v_{i2} \in Y$, and additionally both edges, e_{i1} and e_{i2}, of u_i have the same number of labels k_i. For clarity, we assume that every label in L appears on some edge of G, which does not make BULM any simpler. So, h is equal to the number of labels that appear in G. Moreover, we restrict attention to instances of BULM in which $\sum_i k_i \ge h$, because an instance in which every maximum matching has less than h labels is a trivial "no" instance. Construct a temporal graph $D = (V, A)$ as follows. For every $u_i \in X$ create a cycle of length $2k_i$ which is an alternation of the labels of edges e_{i1} and e_{i2}, that is if $l_1, l_2, \dots, l_{k_i}$ and $r_1, r_2, \dots, r_{k_i}$ are the labels of e_{i1} and e_{i2}, respectively, then the edges of the cycle are labeled $l_1, r_1, l_2, r_2, \dots, l_{k_i}, r_{k_i}$ with a single label per edge.

Denote by M' a maximum temporal matching in D. If $|M'| = \sum_i k_i > h$ (where h denotes the number of labels of G), introduce $|M'| - h$ dummy labels $h+1, h+2, \dots, |M'|$ and assign them to every edge of G_D.

[1]The **NP**-completeness of Balanced 3SAT follows by the dichotomy theorem of Schaefer [Sch78], but there is also a straightforward reduction from 3SAT to Balanced 3SAT (described in [ATN12]). The idea is to extend the original 3SAT formula ϕ with additional "dummy" clauses, in a way that makes it balanced without affecting its evaluation. In particular, to obtain a Balanced 3SAT formula ϕ', set $\phi' = \phi$ and repeat the following: for every variable x_i that is still unbalanced in ϕ', improve its balance by adding to ϕ' a new clause of the form $(l_i \vee y \vee \overline{y})$, where l_i is either x_i or $\overline{x_i}$ depending on which of the two had fewer appearances. Observe that all the new clauses are trivially satisfied, because the dummy variable y appears in all of them both negated and non-negated, and that y is balanced.

This completes the reduction. Note that M' takes from each cycle C_i of G_D either all edges corresponding to e_{i1} or all edges corresponding to e_{i2} (one of the two alternating matchings) and then picks for each edge a single label from those available to it.

($\Rightarrow$) If G has a maximum matching M that covers all labels by union, then M corresponds to a selection of either e_{i1} or e_{i2} for every u_i and translates to a maximum matching M' of D by selecting the corresponding maximum matching from every cycle of D. As M covers all labels, it holds that for every label $l \in L$ there is an $e \in M$ s.t. $l \in \lambda(e)$, implying h edges of M' covering the h non-dummy labels (by a single label per edge). We remain with $|M'| - h$ edges each having all $|M'| - h$ dummy labels, so we can assign a different dummy label to each one of them. So, the time-edges of M' have pairwise distinct times. We conclude that M' is a maximum temporal matching of D.

($\Leftarrow$) Take a maximum temporal matching M' of D. Temporality of M' implies that the times of its edges are pairwise distinct. But as the number of distinct times in D is equal to $h + (|M'| - h) = |M'|$, we have that for every possible time there is an edge in M' appearing at that time. Restricting attention to the times corresponding to L, we get that, for every label $l \in L$, M' has an edge appearing at time l. Additionally, M' is maximum, so it must necessarily correspond to a selection of a maximum matching from every cycle of D. From the covering of the first h (non-dummy) labels we have that the corresponding maximum matching of G covers all h labels. $\square$

In several cases, we are interested in computing a temporal matching such that any two consecutive edges in its time-respecting ordering have a time difference of at least k. This is captured by problem Max-TEM($\geq k$). For example, a polynomial-time algorithm for Max-TEM(≥ 2) would translate into a good approximation for TTSP(1,2). The idea is that we would compute such a matching on the subgraph consisting only of the time-edges with cost 1. Then, as the edges of the matching have time differences at least 2, we would connect the edges of the matching in a time-respecting way to form a TTSP tour. The following corollary states that we cannot hope for a polynomial-time algorithm for Max-TEM($\geq k$) so we have to look for approximations.

Corollary 1. *Max-TEM($\geq k$) is* **NP***-hard, for every $k \geq 1$ that is independent of the lifetime and computable in polynomial time.*

Proof. The reduction is from TEM, proved **NP**-complete in Theorem 4. Given an instance $D = (V, A)$ of TEM we construct an instance $D' = (V, A')$ of Max-TEM($\geq k$) as follows. A' is A stretched in time by introducing a $k - 1$ delay between any two consecutive instances of A. Formally, $A'(1 + (t - 1)k) = A(t)$ for $1 \leq t \leq \alpha(D)$ and $A'(t') = \emptyset$ for all other times t'. If D has a perfect temporal matching, then the same matching is in D' a perfect temporal matching with time differences $\geq k$ between its edges. If D' has a perfect temporal matching with time differences $\geq k$, then such a matching can only use edges from the non-empty instances of A' so it corresponds to a perfect temporal matching of D. $\square$

4.2. Approximating TTSP(1,2) via Maximum Independent Sets

Clearly, by taking an arbitrary temporal TSP tour, one obtains a trivial 2-factor approximation for TTSP(1,2). In the worst case, its cost is $2n$ (paying always 2s) while the cost of the optimum TSP tour is at least n (paying always 1s). The immediate question is whether we can do better. Recall that, in the static ATSP(1,2) it is known that we can do much better as there is a (5/4)-factor approximation [Blä04]. In this section, we provide our first (polynomial-time) approximation algorithms for both the generic TTSP(1,2) and its special case with lifetime restricted to n. To do this, we first show that, though by Corollary 1 Max-TEM(≥ 2) is **NP**-hard, it can still be approximated within some constant via a reduction to Maximum Independent Set in 5-claw free graphs. Recall that a graph is k-claw free if there is no k-independent set in the neighborhood of any node. We then translate this to an approximation for TTSP(1,2). For the restricted lifetime case we follow in Section 4.2.1 a similar approach by approximating a temporal path packing this time.

We begin by showing that a constant factor approximation algorithm for Max-TEM(≥ 2) translates to a constant approximation algorithm for TTSP(1,2) strictly smaller than 2. This then naturally motivates us to search for constant approximations for temporal matchings.

Lemma 3. *An $(1/c)$-factor approximation for Max-TEM(≥ 2) implies a $(2-\frac{1}{2c})$-factor approximation for TTSP(1,2).*

Proof. Take any temporal perfect matching M and denote by x the number of edges with cost 1 in it. Clearly, its total cost is $x+2(n/2-x)=n-x$. Denote by OPT_M and ALG_M the optimum solution to Max-TEM(≥ 2) and the solution to Max-TEM(≥ 2) produced by an algorithm, respectively, and by OPT_{TTSP} and ALG_{TTSP} the optimum TTSP tour and the TTSP tour produced by an algorithm, respectively. By assumption we have

$$\mathrm{ALG}_M \geq \frac{1}{c}\mathrm{OPT}_M \tag{1}$$

Now given an instance of TTSP(1,2) disregard all time-edges labeled 2. In this manner we obtain a temporal graph D whose time-edges correspond to edges labeled 1 in the original TTSP(1,2) instance. Denote now by $\mathrm{OPT}_M(1,2)$ and $\mathrm{ALG}_M(1,2)$ the cost of a minimum cost perfect matching with time differences ≥ 2 on the instance of TTSP(1,2) (i.e. one maximizing the number of 1s that it takes) and the cost produced by an algorithm, respectively. Observe that $\mathrm{OPT}_M(1,2)=n-\mathrm{OPT}_M$ as computing a minimum cost temporal matching of the TTSP(1,2) instance is equivalent to computing a maximum temporal matching of D (i.e. the one maximizing the number of 1s taken). We have:

$$\begin{aligned}
\mathrm{ALG}_M(1,2) &= n-\mathrm{ALG}_M \leq n-\frac{1}{c}\mathrm{OPT}_M = n-\frac{1}{c}(n-\mathrm{OPT}_M(1,2))\\
&= n-\frac{n}{c}+\frac{1}{c}\mathrm{OPT}_M(1,2) \leq \frac{c-1}{c}n+\frac{1}{2c}\mathrm{OPT}_{TTSP}\\
&\leq \frac{c-1}{c}\mathrm{OPT}_{TTSP}+\frac{1}{2c}\mathrm{OPT}_{TTSP} = \frac{2c-1}{2c}\mathrm{OPT}_{TTSP}.
\end{aligned}$$

Moreover,

$$\begin{aligned}
\mathrm{ALG}_{TTSP}(1,2) &= \mathrm{ALG}_M(1,2)+n\\
&\leq \frac{2c-1}{2c}\mathrm{OPT}_{TTSP}+\mathrm{OPT}_{TTSP}\\
&= (2-\frac{1}{2c})\mathrm{OPT}_{TTSP}.
\end{aligned}$$

□

We now show that there is, indeed, a constant factor approximation for Max-TEM.

Theorem 5. *There is a $(3/5)$-factor approximation algorithm for Max-TEM.*

Proof. We are given a temporal graph $D=(V,A)$ and our goal is to return a temporal matching of maximum cardinality. To simplify the description let us consider the static expansion $H=(S,E)$ of D (recall, that we always consider static expansions without vertical edges). A set of edges $M\subseteq E$ form a temporal matching of D iff the following conditions are satisfied:

1. Matching Condition: for all $e_1,e_2\in M$ it holds that e_1 and e_2 do not share an endpoint in V.
2. Time-Respecting Condition: for all $(u_{(i_1-1)j_1},u_{i_1j_1'}),(u_{(i_2-1)j_2},u_{i_2j_2'})\in M$ it holds that $i_1\neq i_2$, that is no two edges of the matching appear at the same time.

Now given an edge $e=(u_{(i-1)j},u_{ij'})$ of the static expansion we may think of it as having the following positions for conflicts with other edges, i.e. edges that cannot be taken together with e in a temporal matching:

1. Edges of the same row as e, i.e. all edges of the form $(u_{(i-1)l},u_{il'})$.
2. Edges of the same column as $u_{(i-1)j}$, i.e. all edges that have one endpoint of the form u_{kj}.

3. Edges of the same column as $u_{ij'}$, i.e. all edges that have one endpoint of the form $u_{kj'}$.

Consider now the graph $G = (E, K)$ where $(e_1, e_2) \in K$ iff e_1 and e_2 satisfy some of the above three constraints. Observe that the set of nodes E of G is the set of edges of the static expansion H. It is straightforward to observe that temporal matchings of D are now equivalent to independent sets of G. Observe now that G is 4-claw free which means that there is no 4-independent set in the neighborhood of any node. To see this take any $e \in E$ and any set $\{e_1, e_2, e_3, e_4\}$ of four neighbors of e in G. As there are only three constraints at least two of the neighbors, say e_i and e_j, must be connected to e by the same constraint. Finally, observe that if e_i and e_j both satisfy the same constraint with e (e.g. belong to the same row as e) then they must satisfy the same constraint with each other (e.g. if e_i and e_j belong to the same row as e then e_i belongs to the same row as e_j) implying that e_i and e_j are also connected by an edge in G. From [Hal95] we have that in 4-claw free graphs MAXIMUM INDEPENDENT SET can be approximated within 3/5. □

The following lemma makes a slight modification to the proof of Theorem 5 to obtain a constant approximation for Max-TEM(≥ 2).

Lemma 4. *There is a $\frac{1}{2+\varepsilon}$-factor approximation algorithm for Max-TEM(≥ 2).*

Proof. Recall that a feasible solution now must satisfy the additional constraint that any two time-edges of the temporal matching should differ in time by at least 2, e.g. a feasible solution could be $(e_1, 1), (e_2, 3), (e_3, 5) \ldots, (e_k, t_k)$. To take this into account in the reduction to MAXIMUM INDEPENDENT SET, we replace constraint 1, that constraints two edges of the same row, with the following constraint: "A given edge $e = (u_{(i-1)j}, u_{ij'})$ cannot be in a feasible solution with edges of neighboring rows, i.e. all edges $(u_{(k-1)l}, u_{kl'})$ for all $i - 2 \leq k - 1 \leq i$". Now take any 3 edges that are constrained to e according to the above constraint. If any two of them are in the same row then they are also constrained with each other so we can have at most 2 independent edges. Thus, assume that they are all in different rows, i.e. they are of the form $e_1 = (u_{(i-2)j_1}, u_{(i-1)j_1'})$, $e_2 = (u_{(i-1)j_2}, u_{ij_2'})$, and $e_3 = (u_{ij_3}, u_{(i+1)j_3'})$. Clearly, the pairs (e_1, e_2) and (e_2, e_3) satisfy the above constraint because each of these pairs contains two edges whose time difference is less than 2 and thus we again get by the new constraint at most 2 independent edges in the neighborhood of e. Taken together with the other two constraints of Theorem 5 concerning the columns we conclude that now we can have at most 4 pairwise independent edges in the neighborhood of any edge, that is the resulting G is now 5-claw free. From [Hal95] we have that in $(h+1)$-claw free graphs, for all $h \geq 4$, MAXIMUM INDEPENDENT SET can be approximated within $1/(h/2 + \varepsilon)$. As in our case $h = 4$ we have a $(2+\varepsilon)^{-1}$-factor approximation for MAXIMUM INDEPENDENT SET and thus for Max-TEM(≥ 2). □

Theorem 6. *There is a $(7/4 + \varepsilon)$-factor approximation algorithm for TTSP$(1, 2)$.*

Proof. By Lemma 3, an $(1/c)$-factor approximation for Max-TEM(≥ 2) implies a $(2 - 1/(2c))$-factor approximation for TTSP$(1, 2)$. Lemma 4 gives such an algorithm with $c = 2 + \varepsilon$. So, we have

$$\begin{aligned} 2 - \frac{1}{2c} &= 2 - \frac{1}{2(2+\varepsilon)} \overset{\varepsilon'=2\varepsilon}{=} 2 - \frac{1}{4+\varepsilon'} \\ &= \frac{2(4+\varepsilon') - 1}{4+\varepsilon'} = \frac{7+2\varepsilon'}{4+\varepsilon'} \\ &\leq \frac{7}{4} + \frac{2\varepsilon'}{4} = \frac{7}{4} + \varepsilon = 1.75 + \varepsilon. \end{aligned}$$

That is we obtain a $(7/4 + \varepsilon)$-factor approximation algorithm for TTSP$(1, 2)$. □

4.2.1. Lifetime Restricted to n

We now restrict our attention to temporal graphs with lifetime restricted to n. In this case, we show that an extension of the above ideas provides us with an improved $12/7 \approx 1.71$-factor approximation for TTSP$(1, 2)$. A difference now is that instead of approximating a temporal matching we approximate a temporal path packing.

Lemma 5. *An $(1/c)$-factor approximation for CTPP implies a $(2-\frac{1}{c})$-factor approximation for TTSP$(1,2)$ when the lifetime is restricted to n.*

Proof. First observe that every TTSP tour, including the optimum tour, must necessarily use precisely the time-labels $1, 2, \ldots, n$ because otherwise it cannot cover all nodes in n steps. Denote by OPT_{CTPP} and ALG_{CTPP} the optimum solution to CTPP and the solution to CTPP produced by an algorithm, respectively, and by OPT_{TTSP} and ALG_{TTSP} the optimum TTSP tour and the TTSP tour produced by an algorithm, respectively. By assumption we have

$$\mathrm{ALG}_{CTPP} \geq \frac{1}{c}\mathrm{OPT}_{CTPP} \tag{2}$$

Denote by P the maximal CTPP contained in the optimum TTSP tour. We have $\mathrm{OPT}_{TTSP} = |P| + 2(n-|P|)+1 = 2n-|P|+1 \geq 2n - \mathrm{OPT}_{CTPP}+1 \Rightarrow \mathrm{OPT}_{CTPP} \geq 2n - \mathrm{OPT}_{TTSP}+1$. Thus we get

$$\mathrm{ALG}_{CTPP} \geq \frac{1}{c}(2n - \mathrm{OPT}_{TTSP}+1). \tag{3}$$

Now patch the paths of the solution to CTPP in a time-respecting way to get a solution of TTSP. We have:

$$\begin{aligned}
\mathrm{ALG}_{TTSP} &\leq 2n - \mathrm{ALG}_{CTPP}\\
&\leq 2n - \frac{2n}{c} + \frac{1}{c}\mathrm{OPT}_{TTSP} - \frac{1}{c}\\
&= n(2-\frac{2}{c}) + \frac{1}{c}\mathrm{OPT}_{TTSP} - \frac{1}{c}\\
&\leq (2-\frac{2}{c}+\frac{1}{c})\mathrm{OPT}_{TTSP} - \frac{1}{c}\\
&= (2-\frac{1}{c})\mathrm{OPT}_{TTSP} - \frac{1}{c}.
\end{aligned}$$

□

Observe that insisting on a temporal path packing with consecutive time-labels is crucial for Lemma 5 to hold. The reason is that, any solution to path packing containing a temporal path of length 2 in which the two consecutive edges do not have consecutive time-labels, cannot be contained in a TTSP tour when the lifetime is restricted to n. This is because, for a TTSP tour to complete in precisely n steps, it must hold that all of its edges use consecutive time-labels.

Lemma 6. *There is a $\frac{1}{7/2+\varepsilon}$-factor approximation for CTPP.*

Proof. We directly express a CTPP as an independent set of time-edges in the static expansion. Given an edge $e = (u_{ij}, u_{(i+1)j'})$ we add the following constraints:

1. All edges with tail u_{ik}, for all $1 \leq k \leq n$ (i.e. in the same row as e).
2. All edges of the form $(u_{(i-1)k}, u_{il})$ such that $l \neq j$ (i.e. in the previous row of e whose head does not match the tail of e) or $k = j'$ (a possible 2-cycle).
3. All edges of the form $(u_{(i+1)k}, u_{(i+2)l})$ such that $k \neq j'$ (i.e. in the next row of e whose tail does not match the head of e) or $l = j$ (a possible 2-cycle).
4. All edges (with tails) in $[1, i-2] \cup [i+2, n]$ that have an endpoint adjacent to the tail of e (share that column).
5. All edges (with tails) in $[1, i-2] \cup [i+2, n]$ that have an endpoint adjacent to the head of e (share that column). [2]

[2]Constraints 4 and 5 are particular to the Time-Consecutive version of the TPP problem (i.e. CTPP). Apart from excluding paths that are not node-disjoint, these constraints also guarantee that no two consecutive edges with non-consecutive time-labels are selected in a path.

See Figure 4 for an illustration. We now show that the resulting graph of constraints is $(7+1)$-claw free. First of all, it is immediate to observe that the first three constraints can result in at most 3 pairwise independent edges in the neighborhood of e. The reason is that they constrain e to edges in rows i, $i+1$, and $i+2$ (see the gray horizontal rectangles in Figure 4) and any two edges of the same row are not independent. Next observe that constraint 4 has capacity 2. For contradiction, assume that there are three pairwise independent edges that all have an endpoint in the same column as the tail of e. Clearly, at least two of them have time difference ≥ 2, otherwise the three edges would occupy only two rows thus two of them would share a row and they could not have been pairwise independent. As we have two edges with time difference ≥ 2 that share a column, constraint 4 applies in turn to them contradicting their pairwise independence. Precisely the same argument gives capacity 2 for constraint 5. So, in summary, we have maximum capacity $3+2+2=7$ for pairwise independent edges constrained to e. Again from [Hal95] we have that in $(h+1)$-claw free graphs, for all $h \geq 4$, Maximum Independent Set can be approximated within $1/(h/2+\varepsilon)$. As in our case $h=7$ we have a $[1/(7/2+\varepsilon)]$-factor approximation for Maximum Independent Set and thus for CTPP. □

Theorem 7. *There is a $(12/7+\varepsilon)$-factor approximation algorithm for TTSP$(1,2)$ when the lifetime is restricted to n.*

Proof. By Lemma 5, when the lifetime is restricted to n, an $(1/c)$-factor approximation for CTPP implies a $(2-1/c)$-factor approximation for TTSP$(1,2)$. Lemma 6 gives such an algorithm with $c = 7/2+\varepsilon'$. So, by substitution, we obtain a $(12/7+\varepsilon) \approx 1.71$-factor approximation algorithm for TTSP$(1,2)$ when the lifetime is restricted to n. □

4.3. Improved Approximations for TTSP(1,2) via Set Packing

We now present a different reduction idea, from Max-TEM(≥ 2) to k-Set Packing, that gives improved approximations for TTSP(1,2). We begin from the the generic case, in which the lifetime is not restricted and obtain a 3/5-factor for Max-TEM(≥ 2)that translates to an improved $(17/10+\varepsilon)$-factor for TTSP$(1,2)$. Then, in Section 4.3.1, we focus again on lifetime restricted to n and present an algorithm that exploits this alternative reduction and achieves a factor of $(13/8+\varepsilon)$.

Lemma 7. *There is a $\frac{3}{5+\varepsilon}$-factor approximation algorithm for Max-TEM(≥ 2).*

Proof. We express the temporal matching problem as a 4-Set Packing. Then, from [Cyg13], we have that k-Set Packing can be approximated within $3/(k+1+\varepsilon)$ yielding $3/(5+\varepsilon)$ for $k=4$. In k-Set Packing we are given a family $F \subseteq 2^U$ of sets of size at most k, where U is some universe of elements, and we are asked to find a maximum size subfamily of F of pairwise disjoint sets. Given the temporal graph $D=(V,A)$, we set $U = V \cup \{1,2,\ldots,\alpha(D)\}$. Let, as usual, $H=(S,E)$ be the static expansion of D. Construct now F as follows. For every $(u_{ij}, u_{(i+1)j'}) \in E$ set $F \leftarrow F \cup \{\{u_j, u_{j'}, i-1, i\}\}$. Clearly, $\{u_j, u_{j'}, (i-1), i\} \in 2^U$ because u_j, $u_{j'}$, $i-1$, and i are pairwise distinct elements, thus indeed the constructed $F \subseteq 2^U$. Note that every set contains 4 elements, thus we have created an instance of 4-Set Packing. We prove that there is a temporal matching of size h in D iff there is a packing of F of size h.

($\Rightarrow$) Let $M=(e_1,e_2,\ldots,e_h)$ be such a matching. For every time-edge $e \in M$ say e.g. equal to $(u_{ij}, u_{(i+1)j'})$ we have the corresponding set $S_e = \{u_j, u_{j'}, i-1, i\} \in F$. Consider the set $S = \bigcup_{e \in M} S_e$. Clearly, $S \subseteq F$ and $|S| = h$. It remains to show that for all $S_e, S_{e'} \in S$, S_e and $S_{e'}$ are disjoint sets. Let them be $\{u_e, v_e, i_e - 1, i_e\}$ and $\{u_{e'}, v_{e'}, i_{e'}-1, i_{e'}\}$. As M is a temporal matching it must clearly hold that u_e, v_e, $u_{e'}$, and $v_{e'}$ are pairwise distinct. Obviously, the nodes of one set are pairwise distinct to the times of the other set. Finally, observe that the is correspond to the tails of the edges in the static expansion and as we have a ≥ 2 temporal matching it must hold that $|i_{e'} - i_e| > 1$. Assume w.l.o.g. that $i_{e'} > i_e + 1$. Then it follows that $i_{e'} > i_e$, $i_{e'} > i_e - 1$, $i_{e'} - 1 > i_e$, $i_{e'} - 1 > i_e - 1$. Thus, also the times of the two sets are pairwise distinct and we conclude that S_e and $S_{e'}$ are disjoint sets.

($\Leftarrow$) Let $P = (S_1, S_2, \ldots, S_h)$ be such a packing. For every set $S \in P$ say e.g. equal to $\{u_j, u_{j'}, i-1, i\}$ we have the corresponding edge $e_S = \{u_j, u_{j'}, i-1, i\} \in E$. Consider the set $M = \bigcup_{S \in P} e_S$. As every set of F corresponds to a distinct edge of E, M corresponds to h distinct edges of E, i.e. $|M| = h$. Now take

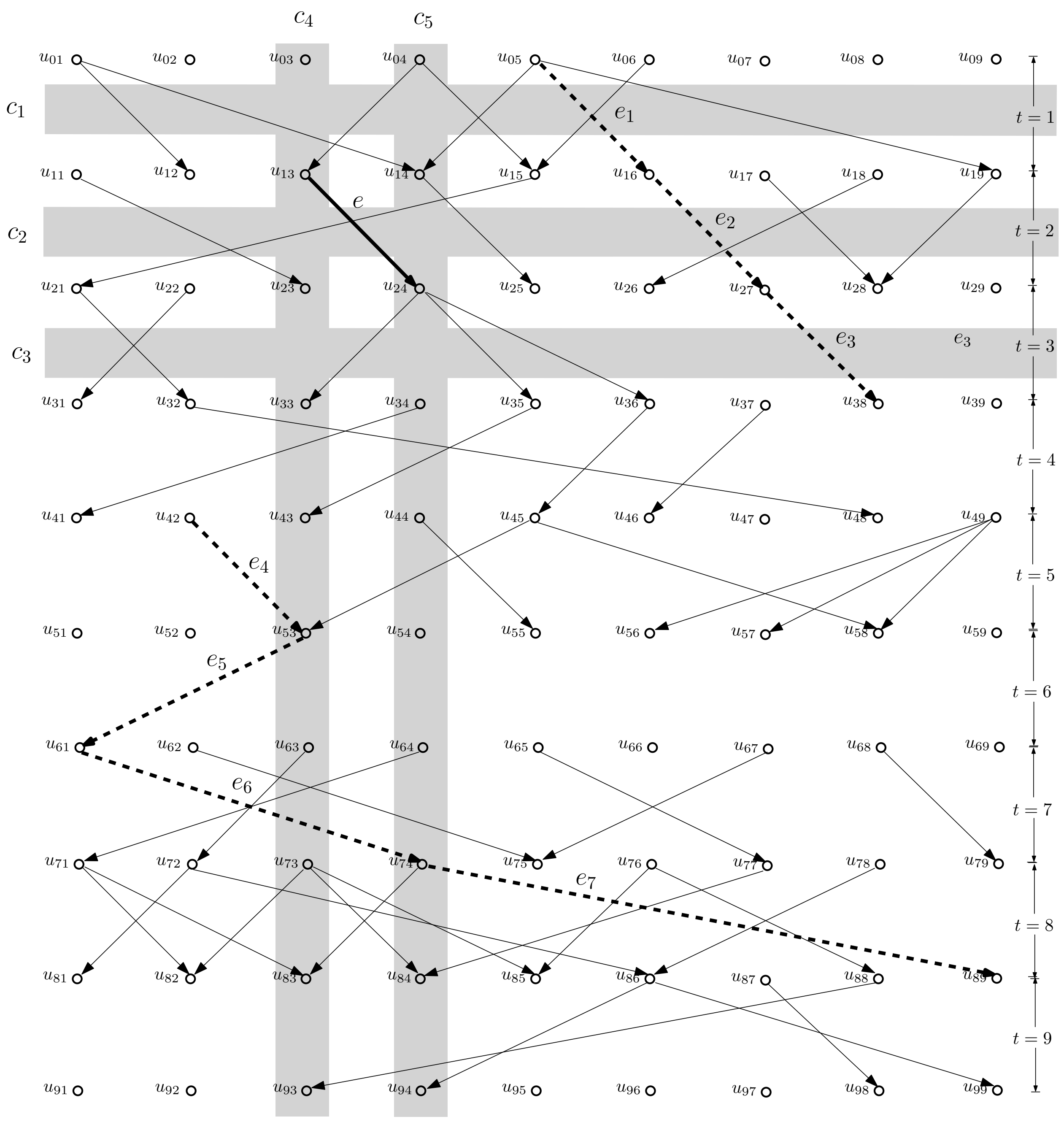
c_4
c_5
c_1
c_2
c_3
e
e_1
e_2
e_3
e_3
e_4
e_5
e_6
e_7
u_{01} u_{02} u_{03} u_{04} u_{05} u_{06} u_{07} u_{08} u_{09}
u_{11} u_{12} u_{13} u_{14} u_{15} u_{16} u_{17} u_{18} u_{19}
u_{21} u_{22} u_{23} u_{24} u_{25} u_{26} u_{27} u_{28} u_{29}
u_{31} u_{32} u_{33} u_{34} u_{35} u_{36} u_{37} u_{38} u_{39}
u_{41} u_{42} u_{43} u_{44} u_{45} u_{46} u_{47} u_{48} u_{49}
u_{51} u_{52} u_{53} u_{54} u_{55} u_{56} u_{57} u_{58} u_{59}
u_{61} u_{62} u_{63} u_{64} u_{65} u_{66} u_{67} u_{68} u_{69}
u_{71} u_{72} u_{73} u_{74} u_{75} u_{76} u_{77} u_{78} u_{79}
u_{81} u_{82} u_{83} u_{84} u_{85} u_{86} u_{87} u_{88} u_{89}
u_{91} u_{92} u_{93} u_{94} u_{95} u_{96} u_{97} u_{98} u_{99}
$t = 1$
$t = 2$
$t = 3$
$t = 4$
$t = 5$
$t = 6$
$t = 7$
$t = 8$
$t = 9$

Figure 4: The bold solid edge $e = (u_{13}, u_{24})$ is the edge under consideration. The bold dashed edges are edges that are constrained to e but not to each other, i.e. they form a 7-claw. The gray rectangles respresent the regions that have some constraint to e. For example, c_2 corresponds to constraint 1 of Lemma 6 and c_1 to 2. Observe that no more edges can be constrained to e and at the same time be independent to all other edges $e_1, e_2, \ldots, e_7$, so there can be no 8-claw in the neighborhood (of constraints) of e.

any two edges in M, say $e = \{u_e, v_e, i_e - 1, i_e\}$ and $e' = \{u_{e'}, v_{e'}, i_{e'} - 1, i_{e'}\}$. Clearly, u_e, $u_{e'}$, v_e, and $v_{e'}$ are pairwise distinct because S_e and $S_{e'}$ are pairwise disjoint, thus M is a matching. Finally, let w.l.o.g. $i_e \leq i_{e'}$. If $i_e \geq i_{e'} - 1$ then either $i_e = i_{e'} - 1$ or $i_e = i_{e'}$. In both cases we obtain a contradiction because if $i_e = i_{e'} - 1$ then the 4th position of S_e is equal to the 3rd position of $S_{e'}$ and if $i_e = i_{e'}$ then the 4th positions of both sets are equal but S_e and $S_{e'}$ are pairwise disjoint. Thus $i_{e'} > i_e + 1$ for all pairs of edges in M and we conclude that M is a temporal matching of size h. □

Theorem 8. *There is a $(1.7 + \varepsilon)$-factor approximation algorithm for TTSP$(1, 2)$.*

Proof. By Lemma 3, an $(1/c)$-factor approximation for Max-TEM(≥ 2) implies a $(2 - 1/(2c))$-factor approximation for TTSP$(1, 2)$. Lemma 7 gives such an algorithm with $c = (5 + \varepsilon')/3$. So, by substitution, we obtain a $1.7 + 2\varepsilon'/(10 + 2\varepsilon') = (1.7 + \varepsilon)$-factor approximation algorithm for TTSP$(1, 2)$. □

4.3.1. Lifetime Restricted to n

Now assume again that the lifetime of the temporal graph is restricted to n. In this case, we devise via a reduction to 3-Set Packing an improved $13/8 = 1.625$-factor approximation for TTSP$(1, 2)$.

Theorem 9. *There is a $(13/8 + \varepsilon)$-factor approximation algorithm for TTSP$(1, 2)$ when the lifetime is restricted to n.*

Proof. Recall that every TTSP tour, including the optimum tour, must necessarily use precisely the time-labels $1, 2, \ldots, n$, because otherwise it cannot cover all nodes in n steps. So, the optimum TTSP tour can be partitioned into two temporal matchings, M_O and M_E, both with time differences ≥ 2 between consecutive labels. M_O is the *odd matching* using labels $1, 3, 5, \ldots$ and M_E is the *even matching* using labels $2, 4, 6, \ldots$. So, if we denote by OPT_{TTSP} the cost of the optimum TTSP tour and by $c(D')$ and $o(D')$ the cost and the number of edges of cost one, respectively, of a single-label (i.e. with a single label per edge) subgraph D' of the temporal graph D, we have

$$\begin{aligned} &\text{OPT}_{TTSP} = c(M_O) + c(M_E) = 2n - [o(M_O) + o(M_E)] \Rightarrow \\ &o(M_O) + o(M_E) = 2n - \text{OPT}_{TTSP} \end{aligned} \tag{4}$$

We now approximate the maximum odd and maximum even matchings of the temporal graph D (counting the number of edges of cost one). Assume, for example, that we want to approximate the maximum matching that uses only odd labels (the even labels case is symmetric). We express it as a 3-Set Packing as follows. Recall that in 3-Set Packing we are given a family $F \subseteq 2^U$ of sets of size at most 3, where U is some universe of elements, and we are asked to find a maximum size subfamily of F of pairwise disjoint sets. We set $U = V \cup L_O$, where $L_O = \{1, 3, 5, \ldots\} \subset \{1, 2, \ldots, n\}$ is the set of all odd labels. Now consider the subgraph $H = (S, E)$ of the static expansion of D consisting only of the edges of cost one that appear at odd times and construct F as follows. For every $(u_{ij}, u_{(i+1)j'}) \in E$ set $F \leftarrow F \cup \{\{u_j, u_{j'}, i\}\}$. Clearly, $\{u_j, u_{j'}, i\} \in 2^U$ because u_j, $u_{j'}$, and i are pairwise distinct elements, thus indeed the constructed $F \subseteq 2^U$. Note that every set contains 3 elements, thus we have created an instance of 3-Set Packing. It is not hard to show that there is an odd temporal matching of size h iff there is a packing of size h. The reason is that two sets $\{u, v, t\}$ and $\{u', v', t'\}$ do not conflict and can be picked together in the packing iff the corresponding edges can be picked at the same time in an odd temporal matching. Now, from [Cyg13], we have that k-Set Packing can be approximated within $3/(k + 1 + \varepsilon)$ yielding $3/(4 + \varepsilon')$ for $k = 3$. We omit ε' in the sequel and add it in the end. So, if we denote by OPT_O and ALG_O (OPT_E and ALG_E) the size of the optimum odd (even) matching and of the odd (even) matching produced by the above algorithm, respectively, we have

$$\text{ALG}_O \geq \frac{3}{4}\text{OPT}_O \tag{5}$$

$$\text{ALG}_E \geq \frac{3}{4}\text{OPT}_E \tag{6}$$

Now from the two computed matchings we keep the one with maximum cardinality. Denote its cardinality by ALG_M. Clearly, $2\mathrm{ALG}_M \geq \mathrm{ALG}_O + \mathrm{ALG}_E$, so we have

$$\begin{aligned}
\mathrm{ALG}_M &\geq \frac{1}{2}(\mathrm{ALG}_O + \mathrm{ALG}_E) \\
&\overset{(5),(6)}{\geq} \frac{1}{2} \cdot \frac{3}{4}(\mathrm{OPT}_O + \mathrm{OPT}_E) \\
&= \frac{3}{8}(\mathrm{OPT}_O + \mathrm{OPT}_E) \\
&\overset{(4)}{\geq} \frac{3}{8}[o(M_O) + o(M_E)] \\
&= \frac{3}{8}(2n - \mathrm{OPT}_{TTSP}) \\
&= \frac{6}{8}n - \frac{3}{8}\mathrm{OPT}_{TTSP}
\end{aligned}$$

So,

$$\mathrm{ALG}_M \geq \frac{6}{8}n - \frac{3}{8}\mathrm{OPT}_{TTSP}. \tag{7}$$

Now, we complete the produced matching arbitrarily with the missing edges to obtain a TTSP tour. Denote by ALG_{TTSP} the cost of the produced TTSP tour. Note that this completion is always feasible as the computed matching has time difference ≥ 2 between its edges. As in the worst case, every added edge has cost 2, we have

$$\begin{aligned}
\mathrm{ALG}_{TTSP} &\leq 2n - \mathrm{ALG}_M \\
&\overset{(7)}{\leq} 2n - \frac{6}{8}n + \frac{3}{8}\mathrm{OPT}_{TTSP} \\
&= \frac{10}{8}n + \frac{3}{8}\mathrm{OPT}_{TTSP} \\
&\leq \frac{10}{8}\mathrm{OPT}_{TTSP} + \frac{3}{8}\mathrm{OPT}_{TTSP} \\
&= \frac{13}{8}\mathrm{OPT}_{TTSP}.
\end{aligned}$$

We conclude that

$$\mathrm{ALG}_{TTSP} \leq \frac{13}{8}\mathrm{OPT}_{TTSP}.$$

Note that $\frac{13}{8} = 1.625$. □

5. Some Further Results

Consider now the Max-TTSP problem in which time-edges have general nonnegative weights and we want a maximum weight TTSP tour.

Theorem 10. *There is a $(1/5)$-factor approximation algorithm for Max-TTSP.*

Proof. Denote by OPT_{MTTSP} (ALG_{MTTSP}) the optimum (found by an algorithm, resp.) Max-TTSP tour and by $\mathrm{OPT}_{TEM(\geq 2)}$ ($\mathrm{ALG}_{TEM(\geq 2)}$) the maximum (found by an algorithm, resp.) weight temporal matching with time differences ≥ 2. First of all, observe that as in Lemma 4 the graph of constraints for the temporal matching is 5-claw free. Now additionally, for every time-edge we have a non-negative weight so we are seeking for a maximum weight independent set in d-claw free graphs, for which there is a $(2/d)$-approximation algorithm [Ber00]. So, by this reduction we get $\mathrm{ALG}_{TEM(\geq 2)} \geq (2/5)\mathrm{OPT}_{TEM(\geq 2)}$. Moreover, observe that the maximum weight TTSP tour may be partitioned into two temporal matchings each of which has weight at most $\mathrm{OPT}_{TEM(\geq 2)}$. Thus, $\mathrm{OPT}_{MTTSP} \leq 2\mathrm{OPT}_{TEM(\geq 2)}$ and $\mathrm{ALG}_{TEM(\geq 2)} \geq (2/5)(\mathrm{OPT}_{MTTSP}/2) = (1/5)\mathrm{OPT}_{MTTSP}$. Finally, observe that $\mathrm{ALG}_{MTTSP} \geq \mathrm{ALG}_{TEM(\geq 2)}$ because connecting the edges of the matching in an arbitrary time-respecting way can only contribute weight. So, we conclude that

$$\mathrm{ALG}_{MTTSP} \geq \frac{1}{5}\mathrm{OPT}_{MTTSP}.$$

$\square$

Remark 3. *To appreciate the expressive power of the labels in temporal graphs consider the following problem: "Given a temporal graph D find a maximum cardinality matching M such that additionally, for all $e_i, e_j \in M$, e_i and e_j never appear at the same time, i.e. no two of them have a label in common". Call it* Maximum Temporal Independent Matching *(TIM). Now take any instance $G = (V, E)$ of* Maximum Independent Set*. Construct the following temporal graph $D = (V', A)$. For every $u_i \in V$ create an edge (u_i, v_i) in G_D so that G_D is just a matching of n edges. For all $(u_i, u_j) \in E$ (the nodes of the edge ordered lexicographically) add the label u_iu_j on edges (u_i, v_i) and (u_j, v_j) of G_D. Observe now that this very special case of TIM is equivalent to* Maximum Independent Set *so there can be no $\frac{1}{n^{1-\varepsilon}}$-factor approximation for any $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$ [Zuc07].*

Theorem 11. *There is some $\varepsilon > 0$, such that kTPP cannot be approximated within k^{ε}, for all $k \geq 3$, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. We show that kTPP is as hard (w.r.t. approximability) as the Maximum Independent Set problem restricted to graphs of degree upper bounded by k. Note that the latter problem has been shown in [AFWZ95] not approximable within k^{ε} (for some $\varepsilon > 0$). Take any instance $G = (V, E)$ of MIS of degree $\leq \Delta$. We will construct a temporal graph D based on G. Informally, for every $u \in V$ we will create a path of length Δ in the static expansion of D "representing" the edges incident to u in G. All paths will be pairwise time-disjoint but two paths will not be node-disjoint iff two of their nodes correspond to the same edge in G, that is we exclude the possibility to pick in a solution two paths that correspond to two nodes that are neighbors in G. The formal construction follows:

- The static expansion of D will consist of $n(\Delta + 1)$ rows (i.e. its lifetime). Moreover, for every edge uv in G the static expansion of D will have an *edge*-column uv and for every node u in G with degree $d(u) < \Delta$ the static expansion will have $\Delta - d(u)$ additional *dummy*-columns $u1, u2, \ldots, u(\Delta - d(u))$. Clearly, the number of columns is $O(\Delta n)$.

- Order the nodes according to their indices in G, as $u_1, u_2, \ldots, u_n$. Node u_i will own the time-window $[(i-1)\Delta + i, i\Delta + (i-1)]$ (of length Δ) and, at times $i\Delta + i$, D will be empty. Next, for every node $u_i \in V$ with neighbors $v_1, v_2, \ldots, v_{d(u_i)}$ in G, create, in the static expansion of D, the path $u_iv_1, u_iv_2, \ldots, u_iv_{d(u)}$ in the first $d(u_i)$ slots of u_i's time-window and if $d(u_i) < \Delta$ append to it the path $u_i1, u_i2, \ldots, u_i(\Delta - d(u_i))$ drawn in the remaining time-slots.

Assume that G has an independent set S of size l. Then, for every $u, v \in S$, the paths of u and v are trivially time-disjoint, because by construction each path has its own time-slot, but also they are node-disjoint as follows. If they were not, then this means that they share a column. This cannot be one of the dummy-columns ui or vj because each of the nodes could only use its own dummy columns. So it must be an edge-column wz. But all edge-columns visited by the path of u satisfy that either w of z is equal to u and all edge-columns visited by the path of v satisfy that either w or z is equal to v. It follows that both can only be satisfied (given that $u \neq v$) if $wz = uv$, which in turn implies that u and v are neighbors in G. But the latter is a contradiction, because u and v belong to the independent set S of G.

For the other direction, assume that D has l node-disjoint paths of length Δ (which are also always time-disjoint). Pick the set of l nodes $S \subseteq V$ corresponding to the nodes owning the paths. Assume that there are $u, v \in S$ that are neighbors in G. This implies that both their paths go through column uv of the static expansion of D, violating their node-disjointness.

So, the construction satisfies that there are l such paths of length (at least) Δ in D iff there is an independent set of size l in G. By this, OPTs match and objectives of solutions match so kTPP must be at least as hard, in terms of approximability, as Maximum Independent Set with degree bounded by k. $\square$

A natural approach for obtaining in static graphs good bounds on TSP$(1,2)$ is to compute some sort of minimum cost cycle cover (or path packing). This brought us to the previous negative result for temporal path packing and now we go one step further giving a negative result for temporal cycle cover.

In words, we want the cycles to correspond to disjoint intervals so that we can then patch them somehow and obtain a TTSP tour. A simpler version, shown below to be approximable within some constant, is just to require the edges of the cycles to have distinct labels, but this does not seem to transform into a good TTSP tour.

We now prove the following theorem stating that Minimum Temporal Cycle Cover (TCC) cannot be approximated assuming $\mathbf{P} \neq \mathbf{NP}$. (Notice the great contrast to the static Minimum Cycle Cover which is solvable in polynomial time via reduction to Minimum Cost Perfect Matching)

Theorem 12. *For any polynomial time computable function $\alpha(n)$, TCC cannot be approximated within a factor $\alpha(n)$, unless $\mathbf{P} = \mathbf{NP}$.*

Proof. The reduction is from the directed Hampath. Take a digraph $G = (V, E)$ and a source node $u_1 \in V$ for the directed hamiltonian path. Create a temporal graph D of lifetime $n = |V|$. For simplicity consider its static expansion. Set $V[D] = V[G]$. Our goal is to force a cheap cycle cover of D to be necessarily a hamiltonian cycle from u_1. Note that since the lifetime is n, a cycle cover must necessarily use all n labels, i.e. have an edge at every row of the expansion.

- Charge all edges (u_{0i}, u_{1j}) for $i > 1$ by $a(n)n$, that is we make it very expensive to begin the cycle cover from some node different than u_1.
- Additionally, charge $a(n)n$ all edges $(u_{ki}, u_{(k+1)1})$, for $k+1 < n$, i.e. all edges going back to u_1 before time n.
- Charge 1 all edges $(u_{(n-1)i}, u_{n1})$, i.e. all edges going back to u_1 at time n.
- Finally, charge all the remaining edges of D as follows: $(u_{ki}, u_{(k+1)j}) = 1$ if $(u_i, u_j) \in E[G]$ otherwise $(u_{ki}, u_{(k+1)j}) = a(n)n$, i.e. all edges of G are cheap in G' and all non-edges expensive.

($\Rightarrow$) If there is a hamiltonian path from $u_1 \in G$ then the time-respecting version of the same path in D together with a final edge going back to u_1 gives a cycle cover of cost n. In this case,

$$\mathrm{OPT}_{TCC} = n.$$

($\Leftarrow$) If there isn't a hamiltonian path from $u_1 \in G$ then we have the following cases:

1. if the cycle cover begins from some node different than u_1 it pays $> a(n)n$ because it uses some expensive edge at the first row.
2. if the cycle cover begins from u_1 but has more than two cycles, this implies that at some time $< n$ it returned to u_1 thus paying one of the expensive edges going back to u_1, i.e. paying again $> a(n)n$.
3. if the cycle cover begins from u_1 and has a single cycle then it is a temporal hamiltonian path (before going back to u_1 at time n) corresponding to a permutation of the nodes of G. As G is not hamiltonian from u_1, it follows that the cycle cover of G' uses at least one non-edge of G, for which it pays $a(n)n$, i.e. its total cost is again $> a(n)n$.

In this case,

$$\mathrm{OPT}_{TCC} > a(n)n.$$

$\square$

On the other hand, consider again a weighted temporal graph with general nonnegative weights and ask for a maximum weight cycle cover in which individual cycles do not necessarily correspond to disjoint intervals and are not even time-respecting (i.e. we just ask for a cycle cover in which labels do not appear twice). In this case, we obtain a (1/2)-factor approximation by reduction to maximum weight independent set in 4-claw free graphs (essentially, two time-edges are constrained if they share heads, if they share tails, or if they share time-label, which forms a 4-claw free graph of constraints) and by exploiting the $(2/d)$-approximation for maximum weight independent set in d-claw free graphs of [Ber00].

6. Conclusions and Further Research

There are many open problems related to the findings of the present work. The first obvious question is whether a (3/2)-factor is within reach either for the general TTSP(1,2) or for the special case with lifetime restricted to n. This could be achieved by some direct approximation algorithm, by some better reduction, or by the harder way of improving the known approximations for Maximum Independent Set in k-claw free graphs or for Set Packing. Another direction that is worth considering is towards approximations for path packings and cycle covers (even for special cases) because these are problems that are very related to traveling salesman problems and have proved particularly useful in approximating (static) TSP(1,2). Moreover, it would be interesting to know how the generic metric TSP problem behaves in temporal graphs. Is there some temporal analogue of triangle inequality (even a different assumption) that would make the problem approximable? Is there some temporal analogue of symmetry (e.g. periodicity) that does not trivialize the temporal dimension of the problem? Of particular interest also seems to us the model in which the lifetime is n and every edge that changes "state" (e.g. cost/availability) remains in the same "state" for at least k steps. For example, $k = 1$ allows for a fully dynamic temporal graph while $k = n$ gives a static graph. So, k in some sense expresses the degree of dynamicity of the graph. We expect here to have interesting trade-offs between k and the approximation factors obtained for temporal problems. Finally, it would be interesting to consider the above and other problems in a special case of temporal graphs defined by the mobility patterns of mobile wireless entities. In this case, the resulting temporal graph would for example have unit disk graphs as instances and it would be interesting to know whether this extra assumption allows for better approximations (we already know it does in static graphs). This direction is also very well motivated as such systems are a common and natural source of temporal graphs and have highly influenced the study of temporal graphs and temporal graph problems.

Acknowledgements. We would like to thank an anonymous reviewer of this article, whose thorough comments have helped us to improve our work.

References

[AAD+06] D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, pages 235–253, March 2006.

[AFWZ95] N. Alon, U. Feige, A. Wigderson, and D. Zuckerman. Derandomized graph products. *Computational Complexity*, 5[1]:60–75, 1995.

[AGM+10] 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. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 379–389, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[AKL08] C. Avin, M. Koucký, and Z. Lotker. How to explore a fast-changing world (cover time of a simple random walk on evolving graphs). In *Proceedings of the 35th international colloquium on Automata, Languages and Programming (ICALP), Part I*, pages 121–132. Springer-Verlag, 2008.

[ATN12] I. Araya, G. Trombettoni, and B. Neveu. A contractor based on convex interval taylor. In *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*, pages 1–16. Springer, 2012.

[Ber96] K. A. Berman. Vulnerability of scheduled networks and a generalization of Menger's theorem. *Networks*, 28[3]:125–134, 1996.

[Ber00] P. Berman. A d/2 approximation for maximum weight independent set in d-claw free graphs. In *Algorithm Theory-SWAT 2000*, pages 214–219. Springer, 2000.

[BK06] P. Berman and M. Karpinski. 8/7-approximation algorithm for (1,2)-TSP. In *Proceedings of the 17th annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 641–648. ACM, 2006.

[BL97] H. Broersma and X. Li. Spanning trees with many or few colors in edge-colored graphs. *Discussiones Mathematicae Graph Theory*, 17[2]:259–269, 1997.

[Blä04] M. Bläser. A 3/4-approximation algorithm for maximum atsp with weights zero and one. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 61–71. Springer, 2004.

[BLWZ05] H. Broersma, X. Li, G. Woeginger, and S. Zhang. Paths and cycles in colored graphs. *Australasian Journal on Combinatorics*, 31:299–311, 2005.

[CFQS12] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *IJPEDS*, 27[5]:387–408, 2012.

[Chr76] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, 1976.

[CMM$^+$08] A. E. Clementi, C. Macci, A. Monti, F. Pasquale, and R. Silvestri. Flooding time in edge-markovian dynamic graphs. In *Proc. of the 27th ACM Symp. on Principles of distributed computing (PODC)*, pages 213–222, 2008.

[Cyg13] M. Cygan. Improved approximation for 3-dimensional matching via bounded pathwidth local search. In *Proceedings of the IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, Washington, DC, USA, 2013. IEEE Computer Society.

[DPR$^+$13] C. Dutta, G. Pandurangan, R. Rajaraman, Z. Sun, and E. Viola. On the complexity of information spreading in dynamic networks. In *Proc. of the 24th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 717–736, 2013.

[Edm65] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17[3]:449–467, 1965.

[FGM82] A. M. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12[1]:23–39, 1982.

[FT98] L. Fleischer and É. Tardos. Efficient continuous-time dynamic network flow algorithms. *Operations Research Letters*, 23[3]:71–80, 1998.

[GSS11] S. O. Gharan, A. Saberi, and M. Singh. A randomized rounding approach to the traveling salesman problem. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 550–559, Washington, DC, USA, 2011. IEEE Computer Society.

[Hal95] M. M. Halldórsson. Approximating discrete collections via local improvements. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 160–169. Society for Industrial and Applied Mathematics, 1995.

[HS12] P. Holme and J. Saramäki. Temporal networks. *Physics reports*, 519[3]:97–125, 2012.

[KKK00] D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC)*, pages 504–513, 2000.

[KLO10] F. Kuhn, N. Lynch, and R. Oshman. Distributed computation in dynamic networks. In *Proceedings of the 42nd ACM symposium on Theory of computing (STOC)*, pages 513–522, New York, NY, USA, 2010. ACM.

[KLS13] M. Karpinski, M. Lampis, and R. Schmied. New inapproximability bounds for tsp. In L. Cai, S.-W. Cheng, and T.-W. Lam, editors, *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC)*, volume 8283 of *Lecture Notes in Computer Science*, pages 568–578. Springer Berlin Heidelberg, 2013.

[KS12] M. Karpinski and R. Schmied. On approximation lower bounds for tsp with bounded metrics. *arXiv preprint arXiv:1201.5821*, 2012.

[KS13] M. Karpinski and R. Schmied. On improved inapproximability results for the shortest superstring and related problems. *Proc. 19th CATS*, pages 27–36, 2013.

[KW98] S. O. Krumke and H.-C. Wirth. On the minimum label spanning tree problem. *Information Processing Letters*, 66[2]:81–85, 1998.

[MCS11a] O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Mediated population protocols. *Theoretical Computer Science*, 412[22]:2434–2450, May 2011.

[MCS11b] O. Michail, I. Chatzigiannakis, and P. G. Spirakis. *New Models for Population Protocols*. N. A. Lynch (Ed), Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool, 2011.

[MCS14] O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Causality, influence, and computation in possibly disconnected synchronous dynamic networks. *Journal of Parallel and Distributed Computing*, 74[1]:2016 – 2026, 2014.

[Mic15] O. Michail. An introduction to temporal graphs: An algorithmic perspective. In *Algorithms, Probability, Networks, and Games - Scientific Papers and Essays Dedicated to Paul G. Spirakis on the Occasion of His 60th Birthday*, Lecture Notes in Computer Science, pages 308–343. Springer, 2015.

[MMCS13] G. B. Mertzios, O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Temporal network optimization subject to connectivity constraints. In *40th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 7966 of *Lecture Notes in Computer Science*, pages 663–674. Springer, July 2013.

[Mon05] J. Monnot. The labeled perfect matching in bipartite graphs. *Information Processing Letters*, 96[3]:81–88, 2005.

[MR02] M. Molloy and B. Reed. *Graph colouring and the probabilistic method*, volume 23. Springer, 2002.

[MS14a] O. Michail and P. G. Spirakis. Simple and efficient local codes for distributed stable network construction. In *Proceedings of the 33rd ACM Symposium on Principles of Distributed Computing (PODC)*, PODC '14, pages 76–85. ACM, 2014.

[MS14b] O. Michail and P. G. Spirakis. Traveling salesman problems in temporal graphs. In *39th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 553–564. Springer, 2014.

[Orl81] J. B. Orlin. The complexity of dynamic languages and dynamic optimization problems. In *Proceedings of the 13th annual ACM symposium on Theory of computing (STOC)*, pages 218–227. ACM, 1981.

[PY91] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84[1]:127–150, 1991.

[PY93] C. H. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18[1]:1–11, 1993.

[Sch78] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th annual ACM symposium on Theory of computing (STOC)*, pages 216–226. ACM, 1978.

[Sch02] C. Scheideler. Models and techniques for communication in dynamic networks. In *Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 27–49, 2002.

[SV14] A. Sebő 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*, 5[34]:597–629, 2014.

[TIR78] S. L. Tanimoto, A. Itai, and M. Rodeh. Some matching problems for bipartite graphs. *Journal of the ACM (JACM)*, 25[4]:517–525, 1978.

[XFJ03] B. Xuan, A. Ferreira, and A. Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14[02]:267–285, 2003.

[Zuc07] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory Of Computing*, 3:103–128, 2007.