Temporal Connectivity Augmentation

Thomas Bellitto
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

Jules Bouton Popper
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

Bruno Escoffier
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

Abstract

Connectivity in temporal graphs relies on the notion of temporal paths, in which edges follow a chronological order (either strict or non-strict). In this work, we investigate the question of how to make a temporal graph connected. More precisely, we tackle the problem of finding, among a set of proposed temporal edges, the smallest subset such that its addition makes the graph temporally connected (TCA). We study the complexity of this problem and variants, under restricted lifespan of the graph, i.e. the maximum time step in the graph. Our main result on TCA is that for any fixed lifespan at least 2, it is NP-complete in both the strict and non-strict setting. We additionally provide a set of restrictions in the non-strict setting which makes the problem solvable in polynomial time and design an algorithm achieving this complexity. Interestingly, we prove that the source variant (making a given vertex a source in the augmented graph) is as difficult as TCA. On the opposite, we prove that the version where a list of connectivity demands has to be satisfied is solvable in polynomial time, when the size of the list is fixed. Finally, we highlight a variant of the previous case for which even with two pairs the problem is already NP-hard.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Mathematics of computing → Paths and connectivity problems

Keywords and phrases Temporal graph, temporal connectivity

Digital Object Identifier 10.4230/LIPIcs.SAND.2025.3

1 Introduction

Temporal graphs are a generalization of graphs, where edges between two nodes can only be used at given time steps. Adding a notion of temporality on the edges makes temporal graphs a powerful tool to study dynamic systems. For example, temporality makes them much more suitable to model public transport networks, where trains between two cities only run at given times, or to study the spread of information or an epidemic [12].

Generalizing notions of graph theory to temporal graphs is an important and active topic but also a challenging one. Indeed, we lose very fundamental properties of accessibility in graph such as symmetry and transitivity: if there is an edge between u and v at time 1 and one between v and w at time 2, it is possible to go from u to w but not from w to u even though one can go from w to v and from v to u. As such, notions such as connected components or spanning trees do not translate well to temporal graphs.

The main topic of this paper is the question of how to (re)connect a temporal graph. Spanners are a well-studied topic in temporal graph theory [16, 3, 1, 8, 9, 2] and can be seen as a generalization of spanning trees. The problem of minimal spanner is, given a connected graph, to remove as many edges as possible without losing the connectivity of the graph. Here, we take a converse approach and start from a graph which is not connected and look for the smallest set of temporal edges to add to make the graph connected. For example, if we want to strengthen a transportation network, which line would be most beneficial for the connectivity of the network. An important variant is to pick the edges we add within a given set of possible edges. For example, if a network is damaged, a solution to this problem would highlight which edges should be repaired first to make the network connected again as quickly as possible.

Using the model of temporal graphs introduced by Kempe et al. [16] where time is discretized into time steps and the edge set can evolve from a time step to the next one, we formalize our problem as follows: given a temporal graph $\mathcal{G}$ and a set F of temporal edges (not in $\mathcal{G}$), we want to determine a set $F' \subseteq F$ of minimal size such that adding F' to $\mathcal{G}$ restore some connectivity requirements (see Section 2 for formal definitions).

Our contribution. Our work focuses on the computational complexity of the problems. We look at three different connectivity requirements: in the first one, we want to make the graph temporally connected by addition of temporal edges. We denote the problem by TCA, for temporal connectivity augmentation. In the second one, we fix a specific vertex v and we want it to be a source of the temporal graph (we call this problem TSA, for temporal source augmentation). In the last one, we are given a set of pairs (u_i, v_i) of vertices, and we want that a path exists from u_i to v_i for all i (TPCA, temporal pair connectivity augmentation).

We mainly show that TCA and TSA are NP-hard even when there are only 2 time steps, while TPCA is polynomially solvable when the number of pairs is fixed. We refine the hardness results by showing that TCA and TSA remain hard under other restrictions. We also provide some additional complementary results, mainly a specific case where TCA is polynomially solvable, and a natural variation of TPCA which turns out to be NP-hard even with a fixed number of pairs.

Note that most of these results hold both in the strict and in the non-strict settings, corresponding to the case where temporal paths are required to be strict (involving temporal edges with strictly increasing time) or not.

The article is organized as follows. Formal definitions and preliminaries are given in Section 2. Section 3 present results on TCA and TSA, while Section 4 deals with TPCA.

Further related work. On the topic of temporal connectivity, introduced by Kempe et al. [16], the work of Bhadra and Ferreira [4] focuses on extending the definition of connected components to temporal graphs. Furthering the study of connected components, Jarry and Lotker [15] focus on temporal graphs with geometric properties. Connectivity was also investigated in the time windows setting, for example by Kuhn et al. [17]. In this work, the graph has to be connected for every time interval of length δ. A connectivity problem studied by Michael et al. [18] resembles what our work focuses on. They have a swarm of robots and ask that over time, they maintain connectivity by distributed protocols. On the other side, our approach is centralized, does not start with a connected graph necessarily and only aims at connectivity on the full lifespan of the graph. A framework of analysis has been proposed by Casteigs et al. [7], establishing relevant restrictions to the graph when studying connectivity and a hierarchy between them for complexity results.

As mentioned earlier, another area of interest in temporal graphs is the study of spanners, i.e. subgraphs that preserve connectivity. It is first introduced by Kempe et al. [16], as an open question about finding a sparse subset of edges in an already connected graph. A negative answer was given by Axiotis and Fotakis [3], showing that there are minimal connected temporal graphs with $\Theta(n^2)$ edges. Additionally, they investigate weaker connectivity requirements, as the total connectivity is sometimes not relevant in practical applications. Our work also considers such generalizations.

In [1], Akrida et al. restricted the problem to temporal cliques, i.e. temporal graphs with a complete underlying graph, with unique time labels, but did not improve the asymptotic density of spanners. Furthering the work, in 2021, Casteigts et al. [8] proved that temporal cliques admit sparse spanners with $O(n \log n)$ edges, and designed an algorithm to find such a spanner. The question of whether a linear spanner exists in all temporal cliques remains open. A study of the problem on random temporal graphs is done by Casteigts et al. in [9] to establish sharp thresholds on connectivity properties and the implications for the existence of different kinds of sparse spanners. In [2] Angrick et al. make a considerable step forward by proving the existence of linear spanners in some families of temporal cliques, leaving very constrained cases where the question is still open. Other research on spanners include robustness [5], or inefficient networks [11].

2 Preliminaries and notation

This section introduces important definitions and makes some basic observations that contextualize our problem.

▶ **Definition 1** (temporal graph). *A temporal graph is a pair (G, λ) where $G = (V, E)$ is a (static) graph and $\lambda : E \to 2^{\mathbb{N}}$. $\lambda(e)$ represents the set of time steps where edge e is present. The lifespan of the graph is $T = \max_{e \in E} \max \lambda(e)$. (G, λ) is simple if $|\lambda(e)| = 1$ for all $e \in E$.*

A temporal graph is also classically defined using temporal edges. A temporal edge is a pair (e, t) with $e \in \binom{V}{2}$ and $t \in \mathbb{N} \setminus \{0\}$. Then the temporal edges of (G, λ) are the temporal edges (e, t) where $e \in E$ and $t \in \lambda(e)$.

The snapshot of (G, λ) at time t is the (static) graph $G_t = (V, E_t)$ where E_t is the set of edges $e \in E$ with $t \in \lambda(e)$ (i.e., temporal edges at time t). We will call *snapshot components* at time t the set of connected components of the snapshot at time t.

▶ **Definition 2** (journey). *A journey (or temporal path) of a temporal graph $\mathcal{G}$ is a sequence $(u_0, u_1, t_0), (u_1, u_2, t_1), \ldots, (u_{k-1}, u_k, t_{k-1})$ where:*

- *$(\{u_i, u_{i+1}\}, t_i)$ is a temporal edge of $\mathcal{G}$;*
- *$t_i \le t_{i+1}$ (resp. $t_i < t_{i+1}$) if the journey is non-strict (resp. strict).*

▶ **Definition 3** (temporal connectivity). *A vertex v is reachable from u if there is a journey from u to v (denoted $u \to v$). If every vertex v is reachable from every other vertex u in $\mathcal{G}$, then $\mathcal{G}$ is temporally connected. If all journeys are strict, then the graph is strictly connected.*

▶ **Definition 4** (augmentation). *Let $\mathcal{G}$ be a temporal graph, and F be a set of temporal edges. The augmentation of $\mathcal{G}$ by F is the temporal graph denoted $\mathcal{G} \uparrow F$ obtained from $\mathcal{G}$ by adding the temporal edges in F. We say that F is a (strictly) connecting set for $\mathcal{G}$ if $\mathcal{G} \uparrow F$ is (strictly) connected.*

Now we can formally define the main problems under consideration.

Temporal Connectivity Augmentation (TCA)
Input: A temporal graph $\mathcal{G}$, a set F of temporal edges and an integer K.
Question: Is there set $F' \subseteq F$ of size at most K such that $\mathcal{G} \uparrow F'$ is temporally connected?

As mentioned in introduction, we also consider other connectivity requirements. More precisely, we consider:

- Temporal Source Augmentation (TSA) where, given a specific vertex v, we want to make v a source (by augmentation).

- Temporal Pairs Connectivity Augmentation (TPCA), which takes as additional input a list of pairs of vertices that need to be connected in the augmented graph. Formally:

  Temporal Pairs Connectivity Augmentation (TPCA)
  Input: A temporal graph $\mathcal{G}$, a set F of temporal edges, a set $X = \{(u_1, v_1), \dots, (u_p, v_p)\}$ of p connectivity demands and an integer K.
  Question: Is there a set $F' \subseteq F$ of size at most K such that in $\mathcal{G} \uparrow F'$ there exists a journey from u_i to v_i (for $i = 1, \dots, p$)?

TCA, TSA and TPCA are defined using non strict journeys. We consider also the strict versions, with strict connectivity requirements, namely Strict TCA, Strict TSA and Strict TPCA. To avoid some possible confusion, we will sometimes explicitely precise that TCA, TSA, TPCA refer to the non strict version (and then write Non Strict TCA,...).

We will study the complexity of these problems under several possible restrictions. We will consider the *simple case* when $\mathcal{G} \uparrow F$ is simple, the case where the lifespan of $\mathcal{G} \uparrow F$ is bounded by k (denoted k-TCA, k-TSA, ...), and the *unrestricted case* where any temporal edge can be added (on the existing set of vertices and within the lifespan of the graph), i.e., $F = \{(\{u, v\}, t) : u, v \in V, 1 \leq t \leq T\}$.

To conclude this section we formally state the connection between spanners and TCA mentioned in introduction. The temporal spanner problem asks, given a connected temporal graph and an integer K, if we can maintain connectivity by keeping only K temporal edges of the temporal graph.

▶ **Proposition 5.** *There is a polynomial-time reduction from the temporal spanner problem to the temporal connectivity augmentation problem.*

Proof. Let $I = (\mathcal{G}, K)$ be an instance of the temporal spanner problem. We consider a temporal graph $\mathcal{G}'$ on the same vertex set as $\mathcal{G}$, with no temporal edge. We consider as set F (for the augmentation in TCA) all the temporal edges of $\mathcal{G}$. Then, making $\mathcal{G}'$ connected by adding at most K edges of F is equivalent to having a spanner of size at most K in $\mathcal{G}$. ◀

3 Temporal connectivity augmentation

3.1 Strict TCA

The main result of this section is the following theorem, which states that TCA in the strict setting is NP-complete even for temporal graphs of lifespan 2. This remains true even if the temporal graph is simple, and in the unrestricted case. We also show the same results for the source version Strict TSA.

▶ **Theorem 6.** *Strict 2-TCA is NP-complete, even on simple graphs and even in the unrestricted case.*

Proof. The problem is in NP, considering that checking connectivity can be done by $O(n^2)$ path computations each in polynomial time [20]. We make a reduction from Dominating Set. In that problem, given a (static) graph $G = (V, E)$ and a positive integer K, we want to determine if there is a dominating set of size at most K in G, i.e., a subset $V' \subseteq V$ with $|V'| \leq K$ such that for all $u \in V \backslash V'$ there is a $v \in V'$ for which $\{u, v\} \in E$.

Let $I = (G, K)$ be an instance of DOMINATING SET. We construct $\mathcal{G} = (G', \lambda)$ a temporal graph (see figure 1) as follows:

- $V' = V \cup \{x, y\}$

- E' contains edge $\{x, y\}$ and all edges between any two vertices in $V \cup \{y\}$
- $\lambda(\{u, v\}) = \begin{cases} \{2\} & \text{if } \{u, v\} \in E \cup \{\{x, y\}\} \\ \{1\} & \text{else} \end{cases}$

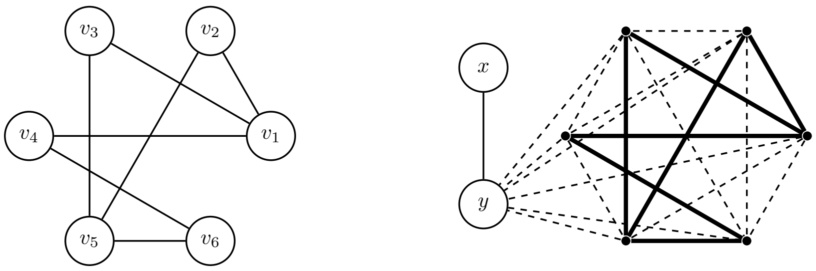


Figure 1 Example of a transformation of a Dominating Set instance into Strict TCA (unrestricted case). Dashed edges are present at time 1, solid edges are present at time 2 and the edges of the original graph are in bold.

As we are in the unrestricted case, any temporal edge can be added. The maximal number of temporal edges that we can add is K.

We now show that there exists a dominating set for $\mathcal{G}$ of size at most K if and only we can make $\mathcal{G}$ temporally connected by adding at most K temporal edges.

Let us first make some remarks. The underlying subgraph of G' excluding x is complete, meaning that this subgraph is already temporally connected (without adding any edge). Moreover, every vertex u of V can reach y at time 1, hence the journey $u \xrightarrow{1} y \xrightarrow{2} x$ exists, so we have $u \to x$ for every $u \in V$. Then, to make $\mathcal{G}$ temporally connected, we have to ensure that $x \to u$ for all $u \in V$.

$\Rightarrow$ Suppose that U is a dominating set for G of size at most K. For each $u \in U$, we add the edge $\{x, u\}$ at time 1. Then each vertex $u \in U$ is reachable from x at time 1. As U is a dominating set, each vertex $v \in V \setminus U$ is adjacent to some vertex $u \in U$. Edge $\{u, v\}$ is by construction present at time 2, so v is reachable from x. The addition of these $|U| \leq K$ temporal edges make the graph connected.

$\Leftarrow$ Conversely, suppose that adding a set of at most K temporal edges makes $\mathcal{G}$ temporally connected. Suppose that we add a temporal edge between two vertices u and v different from x. If it is at time 1 this edge is useless to make x a source. If it is at time 2, this edge is only useful is there is an edge say $\{x, u\}$ or $\{x, v\}$, say $\{x, u\}$, at time 1. Then we can replace the addition of $(\{u, v\}, 2)$ by the addition of $(\{x, v\}, 1)$. This means that adding (only) temporal edges incident to x at time 1 is dominant. Then clearly $(\{x, y\}, 1)$ is useless. So, let U be the set of vertices in V for which a temporal edge $(\{x, u\}, 1)$ has been added. Note that $|U| \leq K$. Take a vertex $v \in V \setminus U$. By construction of $\mathcal{G}$ and definition of U, the unique possible path from x to v is through a vertex $u \in U$ at time 1. Then v must be adjacent to u in V. U is a dominating set of size at most K.

This shows that the unrestricted case is NP-complete. From the above, restricting the augmented set to be in $F = \{(\{(x, v)\}, 1) : v \in V\}$ shows NP-hardness for the simple case. ◀

In the previous reduction, making the graph temporally connected is exactly the same as making x a source of this temporal graph. Thus we derive the following result:

▶ **Corollary 7.** *Strict 2-TSA is NP-complete, even on simple temporal graphs and in the unrestricted case.*

3.2 (Non-strict) TCA

We focus in this section on the non-strict setting. We prove that TCA and TSA are NP-complete on simple graphs of lifespan 2. We first introduce a characterization of connected graphs in the non-strict setting which allows for a reformulation of TCA. We then prove the NP-completeness of this reformulation on simple graphs of lifespan 2. Finally we give a separate proof for the NP-completeness of Simple 2-TSA.

We give the following property $\mathcal{P}$, which is verified when there is a "temporal path" from every snapshot connected components at time 1 to every snapshot connected components at the lifespan of the graph:

▶ **Property** $\mathcal{P}$**.** *Let $C_1^1, \ldots, C_{k_1}^1, \ldots, C_1^T, \ldots, C_{k_T}^T$ be the snapshot components of a temporal graph $\mathcal{G}$. $\mathcal{G}$ satisfies property $\mathcal{P}$ if for every $i_1 \leq k_1$ and $i_T \leq k_T$, there is a sequence of $i_2, \ldots, i_{T-1}$ such that $\forall j < T : |C_{i_j}^j \cap C_{i_{j+1}}^{j+1}| \geq 1$.*

▶ **Theorem 8.** *A temporal graph $\mathcal{G}$ is temporally connected in the non-strict setting if and only if it verifies property $\mathcal{P}$.*

Proof. We prove that $\mathcal{P}$ is a characterization of temporally connected graphs:

$\Rightarrow$ Suppose $\mathcal{G}$ is temporally connected. Take $i_1 \leq k_1$ and $i_T \leq k_T$, and consider $u \in C_{i_1}^1$ and $v \in C_{i_T}^T$. We know that there exists a journey from u to v. Note that by waiting (in u or in v) we can assume without loss of generality that the journey starts (in u) at time 1 and ends (in v) at time T. At each time step t, the journey visits some vertices of a connected component $C_{i_t}^t$ (with $C_{i_1}^1$ containing u and $C_{i_T}^T$ containing v). The last vertex visited in $C_{i_{t-1}}^{t-1}$ is by definition both in $C_{i_{t-1}}^{t-1}$ and $C_{i_t}^t$, so the intersection is not empty.

$\Leftarrow$ Suppose $\mathcal{G}$ verifies $\mathcal{P}$, i.e. for every snapshot component at time 1 and at time T we have a sequence of snapshot components starting at 1 and ending at T, with a non-empty intersection between consecutive components. For any vertex u, we build the journey going to any v with such a sequence. Taking the snapshot component of u at time 1 and the snapshot component of v at time T, and then applying the property $\mathcal{P}$, gives us a sequence between those two components. Since any consecutive snapshot component in that sequence has a non-empty intersection, that means that at every step we can travel to any vertex that is part of that intersection in the snapshot component to get to the next one. Repeating this until v is reached gives us a journey from u to v. ◀

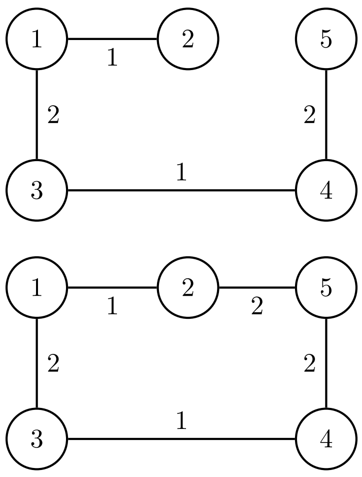
On graphs with a lifespan of 2, we get the following corollary:

▶ **Corollary 9.** *A graph with a lifespan of 2 is temporally connected in the non-strict setting if and only if for every $i_1 \leq k_1$ and every $i_2 \leq k_2$: $|C_{i_1}^1 \cap C_{i_2}^2| \geq 1$.*

In other words, every snapshot component at time 1 has to intersect every snapshot component at time 2.

We can now reformulate the unrestricted version of 2-TCA as a problem on a binary matrix. Let us consider a temporal graph $\mathcal{G}$ of lifespan 2, with k_1 connected components at time 1 and k_2 at time 2. We build a matrix B, that we call the component-intersection matrix of $\mathcal{G}$, with k_1 lines and k_2 columns, such that $B[i, j] = 1$ if C_i^1 and C_j^2 intersect, and 0 otherwise. Adding a temporal edge $(\{u, v\}, 1)$ in $\mathcal{G}$ corresponds to merging the connected components of u and of v at time 1. On matrix B, this corresponds to merging the corresponding rows by an OR-combination. Similarly, adding a temporal edge at time 2 corresponds to merging by an OR-combination the columns corresponding to the components of u and of v at time 2.

In other words, by the construction above the problem can be reformulated as the following one (where combinations can be on rows on or columns).



	$\{1,3\}$	$\{2\}$	$\{4,5\}$
$\{1,2\}$	1	1	0
$\{3,4\}$	1	0	1
$\{5\}$	0	0	1

	$\{1,3\}$	$\{2,4,5\}$
$\{1,2\}$	1	1
$\{3,4\}$	1	1
$\{5\}$	0	1

Figure 2 A graph of life span 2 and its component-intersection matrix, with an example of the augmentation by $\{2, 5\}$ at time 2 corresponding to the merge of columns $\{2\}$ and $\{4, 5\}$.

OR-COMBINATION TO ONE-FILLED (OCTO)
Instance: A binary matrix B and a positive integer K.
Question: Can B be transformed into a one-filled matrix with at most K OR-combinations?

Interestingly, the following lemma shows that the other direction works as well.

▶ **Lemma 10.** *Let B be a binary matrix. There exists a simple temporal graph $\mathcal{G}$ of lifespan 2 such that B is its component-intersection matrix.*

Proof. We put one vertex $v_{i,j}$ for each i, j such that $B[i, j] = 1$. At time 1, we put all edges $\{v_{i,j}, v_{i,j'}\}$ for $j \neq j'$, so that row i corresponds to a connected component at time 1. At time 2, we put all edges $\{v_{i,j}, v_{i',j}\}$ for $i \neq i'$, so that column j corresponds to a connected component at time 2. ◀

Now we prove that OCTO is NP-complete.

▶ **Lemma 11.** *OCTO is NP-complete*

Proof. The problem is trivially in NP. We reduce from the decision version of DISJOINT SET COVERS [6]. In this problem, we are given a finite set $T = \{e_1, \ldots, e_n\}$, a collection $C = \{T_1, \ldots, T_m\}$ of subsets of T and a positive integer K. We want to determine if C can be partitioned into at least K disjoint parts, such that every part is a cover of T.

Let $I = (T, C, K)$ be an instance of DSC. We construct B, a binary matrix with $n(m+1)$ rows and m columns as follows: $B[i, j] = \begin{cases} 1 & e_i \in T_j \\ 0 & e_i \notin T_j \end{cases}, \forall i \in \{1, \ldots, n\}, \forall j \in \{1, \ldots, m\}$. For $i > n$, the matrix repeats itself, $B[i, j] = B[i - n, j]$.

We claim that I is a YES-instance if and only if $I' = (B, m - K)$ is a YES-instance of OCTO.

$\Leftarrow$ Suppose I' is a YES-instance of OCTO. Then there is a set of $p \leq m - K$ OR-combinations of rows or columns that results in a one-filled matrix. Note that the combinations are all on columns, since a meaningful combination on rows has to be done in the $m + 1$ copies of the matrix (thus with $m > m - K$ combinations). After the OR combinations of columns, each of the $m-p \geq K$ remaining columns corresponds to OR-combinations of a set of initial rows: let $O = \{\{c_1^1, \dots, c_{k_1}^1\}, \dots, \{c_1^{m-p}, \dots, c_{k_{m-p}}^{m-p}\}\}$ be the set of combinations done to obtain the columns of the resulting matrix. Since they result from OR combinations, if there is a 1 in some row for some resulting column, that implies that one of the columns used to get that resulting column had a 1 in that row. The matrix is one-filled, so this holds for every resulting column and every row. If we let $\mathcal{C}$ be the same set as O but in which we replace each column by the corresponding set in C, we get a disjoint partition of C, and each element of the partition is a cover for the rows, thus a cover for T. The size is given by $m-p \geq K$, hence I is a YES-instance of DSC.

$\Rightarrow$ Suppose I is a YES-instance of DSC. Let P be a partition of C into $k \geq K$ part, each part being a disjoint covers of T. In the matrix B, for every cover, there is a 1 in each row:

$$\forall j \in \{1, \dots, k\},\ \forall i \in \{1, \dots, n(m+1)\},\ \exists l \in P_j :\ B[i,l] = 1$$

Thus combining the columns in each disjoint cover yields a one-filled matrix. This requires $m-k \leq m-K$ combinations, hence I' is a YES-instance of OCTO. ◀

Together with Lemma 10, from Lemma 11 we derive the following result.

▶ **Theorem 12.** *Non-strict 2-TCA is NP-complete, even on simple graphs and even in the unrestricted case.*

We now look at the source variant in the non-strict setting. The previous result cannot be extended easily like in the previous section; instead we give a separate reduction that proves the following:

▶ **Theorem 13.** *Non-strict 2-TSA is NP-complete, even on simple graphs and even in the unrestricted case.*

Proof. The problem is in NP for the same reason as TCA. We make a reduction from Hitting Set.

Hitting Set
Input: A set $S = \{e_1, \dots, e_n\}$, a collection of subsets $C = \{S_1, \dots, S_m\}, S_i \subseteq S,\ \forall i \in \{1, \dots, m\}$ and a positive integer $K \in \mathbb{N}$ with $K \leq |S|$.
Question: Is there a subset $S' \subseteq S$ with $|S'| \leq K$ such that $S_i \cap S' \neq \emptyset\ \forall i \in \{1, \dots, m\}$?

Let $I = (S, C, K)$ be an instance of Hitting Set. We build a temporal graph $\mathcal{G} = (G, \lambda)$ as follows (an illustration of this graph is shown in Figure 3). Its vertex set V is composed of:

- A vertex x
- For every set $S_j \in C$ that contains the element e_i, a vertex e_iS_j (this set being referred as V_{SC})
- For every set of C, a vertex S_i (this set being referred as V_C)

At time 1, the only present edges are between vertices of V_{SC} such that they correspond to the same element e_i. At time 2, there is an edge between every $e_iS_j \in V_{SC}$ and $S_j \in V_C$.

The reduction is clearly polynomial time. We now prove that there exists a hitting set of S for C with size at most K if and only if there is a set F' of at most K temporal edges such that x is a source in $\mathcal{G} \uparrow F'$.

- $\Leftarrow$ Suppose that F' with $|F'| \leq K$ is such that x is a source in $\mathcal{G} \uparrow F'$. We claim that we can build F'' containing only edges between x and V_{SC} at time 1, such that x is a source in $\mathcal{G} \uparrow F''$. Suppose there is a temporal edge $(\{u, v\}, 1) \in F'$ with u, v different from x. If x cannot reach neither u nor v at time 1, the edge is useless and can also be removed from F'. Otherwise x can reach u or v, say u, at time 1 without using $(\{u, v\}, 1)$. Then $(\{u, v\}, 1)$ may be useful to reach v at time 1, but we can replace $(\{u, v\}, 1)$ by $(\{x, v\}, 1)$ with the same effect. The same reasoning holds for an edge $(\{u, v\}, 2) \in F'$. If x cannot reach neither u nor v at 2 or before, the edge is useless. If x can reach u at some time without using $(\{u, v\}, 2)$, we can replace the edge by $(\{x, v\}, 1)$. After these replacements, we have added only temporal edges of the form $(\{x, v\}, 1)$. To conclude, suppose that we have added an edge $(\{x, S_i\}, 1)$, with $S_i \in V_C$. The only temporal edges incident to S_i are at time 2 (apart from except $(\{x, S_i\}, 1)$), meaning that we do not gain more reachability from x by arriving at time 1 at S_i than at time 2. Then we can safely replace $(\{x, S_i\}, 1)$ by some $(\{x, e_j S_i\}, 1)$ (with $e_j \in S_i$). We build F'' by applying the described replacements to F', and S' by taking the corresponding elements of S appearing as endpoints in F''. Note that $|S'| \leq K$. To reach a vertex S_j, x must use a journey $x \xrightarrow{1} e_i S_j \xrightarrow{2} S_j$ eventually taking multiple edges in the snapshot component at time 1 containing $e_i S_j$. This implies that $e_i \in S_j$, hence $S_j \cap S' \neq \emptyset$. Since every S_j is reachable from x, S' is a hitting set.
- $\Rightarrow$ Conversely, suppose that there exists $S' \subseteq S$, a hitting set with size at most K for C. Then consider F' as the set of edges $(\{x, e_i S_j\}, 1)$ for each $e_i \in S'$, with S_j chosen arbitrarily. Note that $|F'| \leq K$. After augmentation, x can reach every S_j that contains at least one element of S'. Since S' is a hitting set, it follows that x can reach all vertices S_j. Then at time 2 it can reach all remaining $e_i S_j$ from the vertices of V_C. Hence x is a source.

This shows NP-hardness for the unrestricted case. From the above, restricting the augmented set to be included in $F = \{(\{x, v\}, 1), v \in V_{SC}\}$ shows NP-hardness for the simple case. ◀

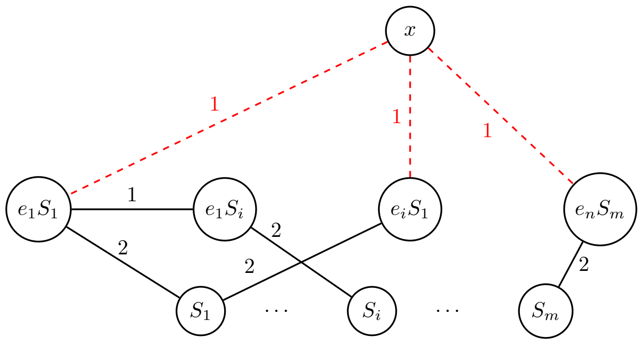


Figure 3 Example of a transformation of an HS instance into 2-TSA. Dashed red edges correspond to the augmentation set for the simple case.

3.3 (1+1)-TCA

We consider in this section the following situation: suppose that we are given a static (non connected) graph, and that are given an extra time slot, $T = 2$, where we can add temporal edges to make the graph connected. The question is to find the minimal number of edges that must be added for this. This is a particular case of TCA (where F is the set of temporal edges at time 2). We show in this section that it is polynomially solvable.

We begin by the following property on connected graphs of lifespan 2.

▶ **Lemma 14.** *Let $\mathcal{G}$ be a temporally connected graph of lifespan 2. Let also $C_1^1, \dots, C_{k_1}^1$ and $C_1^2, \dots, C_{k_2}^2$ be the snapshot components at time 1 and 2. We have $k_1 \leq \min_{i_2 \in \{1,\dots,k_2\}} |C_{i_2}^2|$ and $k_2 \leq \min_{i_1 \in \{1,\dots,k_1\}} |C_{i_1}^1|$.*

Proof. The proof is done by contradiction. Without loss of generality (by symmetry), suppose there are more components at time 1 than the size of the smallest component at time 2. Since the graph is connected, by Lemma 10 each component at time 1 has a non-empty intersection with each component at time 2, including the smallest one. Since there are more snapshot components at time 1 than there are vertices in this smallest component, there are necessarily a vertex in at least two intersections. However, that would imply that it is in both snapshot components at time 1. A vertex in a static graph cannot belong to more than one connected components, hence giving a contradiction. ◀

This gives the core idea for a polynomial time algorithm. We construct at time 2 stars centered at each vertex of the smallest component of the first time step. Every star has at least a branch to all the other components of the fixed time step. The remaining vertices are arbitrarily linked to the first star, see Algorithm 1 for a pseudocode.

Algorithm 1 $(1+1)$-TCA algorithm.

```
Input :
  ▪ A temporal graph $\mathcal{G}$ of lifespan 1
Output :
  ▪ A set $F'$ of temporal edges at time 2 such that $\mathcal{G} \uparrow F'$ is temporally connected.

1 $F' \leftarrow \emptyset$
2 $\ell \leftarrow \arg\min_{i \in \{1,\dots,k_1\}} |C_i^1|$
3 for $i \in \{1,\dots,k_1\} \setminus \{\ell\}$ do
4     for $j \in \{1,\dots,|C_i^1|\}$ do
5         $s \leftarrow (j \mod \ell) + 1$
6         $F' \leftarrow F' \cup \left(\{u_j^{C_i^1}, u_s^{C_\ell^1}\}, 2\right)$
7     end
8 end
9 return $\mu$
```

We give a formal proof of optimality:

▶ **Theorem 15.** $(1+1)$-*TCA is solvable in time* $O(n^2)$.

Proof. The algorithm creates $|C_\ell^1|$ stars that correspond to connected components at time 2. By construction, each star intersects all connected components at time 1, hence by Corollary 9 the graph is temporally connected. At time 2, each connected component $C_{i_2}^2$ has $|C_{i_2}^2| - 1$ edges (since it is a star), which sums to $\sum_{i_2=1}^{|C_\ell^1|} (|C_{i_2}^2| - 1) = n - |C_\ell^1|$. Suppose we add p links at time 2, we have $k_2 \geq n - p$. To be connected we know $k_2 \geq |C_\ell^1|$, where C_ℓ^1 is the smallest component at time 1, hence $p \geq n - |C_\ell^1|$. ◀

4 Temporal pair connectivity augmentation

A natural extension to consider is the problem where instead of all the pairs, we are given a set of pairs to connect by augmentation. As a generalization of TCA and TSA, TPCA is NP-hard both in strict and non-strict settings. We tackle here the problem when the number of pairs is fixed.

4.1 Strict TPCA for a fixed number of pairs

This problem echoes the Directed Generalized Steiner network problem [10], in static graphs:

> DG-Steiner
> **Input:** A directed graph $G = (V, A)$, a weight function $w : A \to \mathbb{R}$, a set $X = \{(u_1, v_1), \ldots, (u_p, v_p)\}$ of p node pairs and two positive integers $B \leq p$ and $K \leq w(\mathcal{G})$.
> **Question:** Is there a subgraph H of G of cost $w(H) \leq B$ that satisfies at least K node pairs from the set X, when a pair (u_i, v_i) is said to be satisfied when there is a path from u_i to v_i?

We will show they are in fact closely related. We employ a similar definition, adapted to temporal graphs:

> TG-Steiner
> **Input:**A temporal graph, $\mathcal{G}$, a weight function w on the temporal edges of $\mathcal{G}$, a set $X = \{(u_1, v_1), \ldots, (u_p, v_p)\}$ of p node pairs and two positive integers $K \leq p$ and $B \leq w(\mathcal{G})$.
> **Question:** Is there a subgraph $\mathcal{H}$ of $\mathcal{G}$ of cost $w(\mathcal{H}) \leq K$ that satisfies at least B node pairs from the set X?

This definition generalizes TPCA: When $B = p$ and the weight assigned to the edges are 0 for edges initially present in the temporal graph and 1 for the edges proposed for augmentation, the problem becomes equivalent.

We now show that the TG-Steiner problem for temporal graphs reduces to DG-Steiner:

▶ **Lemma 16.** *Any instance I of the TG-Steiner problem can be transformed into an instance I' of the DG-Steiner problem in polynomial time.*

Proof. Let $I = (\mathcal{G}, w, X, K, B)$ be an instance of TG-Steiner. Temporal expansion is a classical technique to transform a temporal graph into a static one [19]. We build here a specific expansion that fits our augmentation problem. Let $G_T = (V_T, E_T)$ be (static directed) the graph such that:

- For each vertex v of $\mathcal{G}$ we have $T + 1$ copies v^t, $t = 1, \ldots, T + 1$ in V_T. We also have T arcs (v^t, v^{t+1}), $t = 1, \ldots, T$, of weight 0.
- For each temporal edge $(\{u, v\}, t)$ of weight w of $\mathcal{G}$ we add two vertices $(\{u, v\}, t)$ and $(\{u, v\}, t)'$, with an arc of weight w from the former to the latter. We also add 4 arcs of weight 0: from u^t and from v^t to $(\{u, v\}, t)$, and from $(\{u, v\}, t)'$ to u^{t+1} and to v^{t+1}.

The construction is illustrated in Figure 4. For the set of pairs, if $(u_i, v_i) \in X$ then $(u_i^1, v_i^T) \in X'$, i.e. we replace the source by its earliest copy and the sink by its latest copy. We claim that finding a subgraph H_T of G_T of cost $w_T(H_T) \leq B$ for which at least K node pairs from the set X' are satisfied is equivalent to solving TG-Steiner on I.

$\Rightarrow$ Suppose there exists such a subgraph H_T. Note that the only arcs that have a positive weight are those from $(\{u,v\},t)$ to $(\{u,v\},t)'$. We take in $\mathcal{G}$ all temporal edges $(\{u,v\},t)$ such that $((\{u,v\},t),(\{u,v\},t)')$ is in H_T. In this subgraph of $\mathcal{G}$, which has weight at most K, the connected pairs are the same as those connected in H_T, by construction of our temporal expansion - a path from u_i^1 to v_i^T in H_T corresponds to a journey from u_i to v_i in $\mathcal{G}'$. Hence, we are guaranteed to satisfy the same pairs in H_T and $\mathcal{G}'$.

$\Leftarrow$ Suppose I is a YES-instance, and let $\mathcal{H}$ be a solution. To construct the solution to the DG-Steiner corresponding instance, the previous process is reversed. We take in H_T all the arcs of weight 0, and arcs $((\{u,v\},t),(\{u,v\},t)')$ for which the temporal edges $(\{u,v\},t)$ is selected in $\mathcal{H}$. For the same reasons as before, H_T has the same weight as $\mathcal{H}$ and satisfies the same pairs. ◀

The previous lemma allows us to use the result of [14]:

▶ **Theorem** (Feldman, Ruhl 2006). *p-DG-Steiner is polynomial time solvable when p is a constant. There exists an algorithm solving p-DG-Steiner in* $O(mn^{4p-2} + n^{4p-1}\log n)$.

By noticing the temporal expansion of a temporal graph has $O(nT+M)$ vertices and $O(nT+M)$ arcs, where M denotes the number of temporal edges of the graph, we get through the reduction an algorithm running in time $O((nT+M)^{4p-1}\log(nT+M))$. Hence:

▶ **Theorem 17.** *Strict TPCA is solvable in polynomial time when parameterized by p, the number of pairs to connect.*

4.2 Non-strict TPCA for a fixed number of pairs

Similarly to the strict case, the temporal expansion allows for the use of the result of [14]. The key observation is their result works for any directed graph (not only on DAGs). It is then possible to design a temporal expansion adapted for the non-strict case, taking into account the augmentation part. We achieve this by the following simple addition of arcs: for each pairs of adjacent temporal edges $(\{u,v\},t)$ and $(\{u,w\},t)$ (at the same time step), we add the arcs $((\{u,v\},t)',(\{u,w\},t))$ and $((\{u,w\},t)',(\{u,v\},t))$ in the expansion, see the gray arcs in figure 4. On one hand, this allows paths to travel through multiple edges sharing an endpoint at the same time step. On the other hand, it is impossible to go back in time. As a result, the property that a path in the expansion corresponds to a path in the original graph is preserved. Therefore, we have:

▶ **Theorem 18.** *Non-strict TPCA is solvable in polynomial time when parameterized by p, the number of pairs to connect.*

4.3 Edge-by-edge TPCA

Finally, we study a variation of the problem that we call *edge-by-edge* augmentation. Here, we are given a non-connected temporal graph and a set of possible temporal edges that we may add, but the difference is that if several of these edges have the same endpoints (like $(\{u,v\},t)$ and $(\{u,v\},t')$), we can add all of them at once with a cost of only one.

For example this is what happens if we want to strengthen a transportation network and decide to build a new subway line. By building one line (edge), we add many trains (temporal edges) between its two endpoints.

Since it was the only polynomial case, we first study the TPCA problem under this new setting. We prove that it becomes NP-complete, thus underlying an interesting difference between the two different cost evaluations of an augmentation.

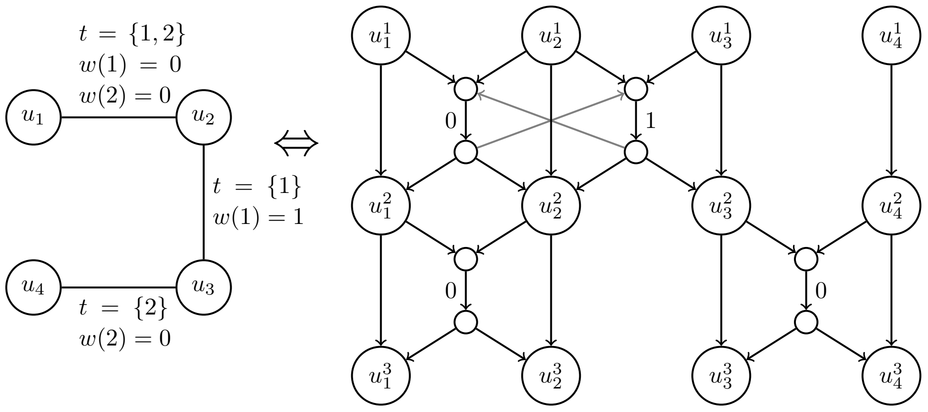


Figure 4 A TG-Steiner instance and its equivalent DG-Steiner expansion. The backward arcs in gray correspond to the non-strict version and the unlabelled arcs have weight 0.

▶ **Theorem 19.** *Non-strict edge-by-edge TPCA is NP-complete even for $p = 2$ and for $T = 2$.*

Proof. The problem is in NP for the same reason as TCA. We make a reduction from 3-SAT. Let $x_1, \dots, x_n$ be the variables and $C_1, \dots, C_m$ the clauses of the statement ϕ. We construct our transformation graph in two parts.
In the variable part of the graph, we concatenate gadgets that encode the truth value of the variable, and all the edges are at time 1. A gadget has a starting vertex, connected to two other vertices, that we call the true and false buffer vertices. In each branch, we alternate between value vertices and buffer vertices. There is a value vertex in each branch for each clause where the variable appears. The value vertex is connected to the last buffer vertex by an optional link, present at times $\{1, 2\}$. The two value vertices are connected to the two following buffer vertices when they exist. The last two value vertices of each branch are connected to the ending vertex of the gadget. The concatenation is done by connecting the ending vertex to the starting vertex of the next variable.
The clause part of the graph has all of its edges at time 2. Like the variable part, the clauses are encoded by a concatenation of gadgets, composed of a starting vertex and an ending vertex. There is an edge for each literal in the clause, from the starting vertex to the buffer preceding the value vertex corresponding to the clause and in the branch corresponding to the literal. There is also an edge from the corresponding value vertex and the ending vertex in the clause part. An example of variable and clause gadgets is shown in figure 5.
We want to connect x_1^S to x_n^E and C_1^S to C_m^E. Remark that a path from x_1^S to x_n^E only takes edges at time 1 and is always of cost $3m$. This is because the only edges with cost 1 are those added for each literal in each clause and there are 3 literals per clause. Moreover, for a valid solution of SAT and in a variable gadget, only the edges of one branch are added, implying that exactly $3m$ edges have been added. They connect x_1^S to x_n^E and C_1^S to C_m^E. Conversely, if there are no solutions with cost $3m$ then the formula cannot be satisfied. ◀

The construction can be adapted for the strict case, however we lose the bound on T. Roughly speaking, the path in the variable part is changed from being at time 1 to being a strictly increasing one ending at α. Then in the clause part, the path at time 2 is changed to a strictly increasing path starting at $\alpha + 1$. This way we preserve the two distinct paths and the proof still holds:

▶ **Theorem 20.** *Strict edge-by-edge TPCA is NP-complete even for $p = 2$.*

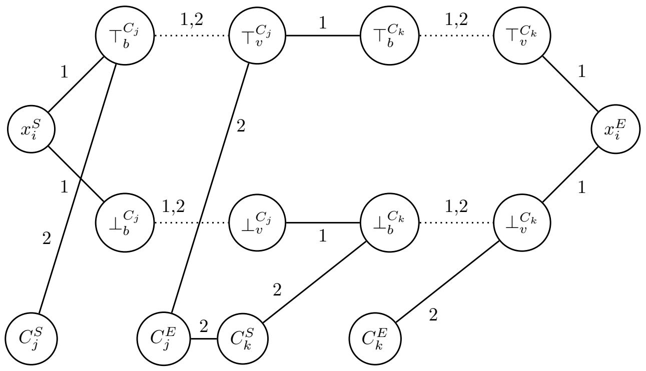


Figure 5 Variable and clause gadgets.

For 2-TCA in the strict setting, we made the observation that only the edges in the augmentation set are interesting to add to the graph. Considering that adding an edge at time 2 is dominated by adding the same edge at time 1, we get that even if we were allowed to add edge-by-edge, the same observation holds. Thus, we get:

▶ **Observation 21.** *Strict EBE 2-TCA is NP-complete.*

As for the non-strict setting, we already establish that the proof is in the unrestricted case by construction. Edge-by-edge would then correspond to allowing doing a column and line merge as one operation, but the optimal solution only does column merges so we would not gain any advantage. It follows then that:

▶ **Observation 22.** *Non-strict EBE 2-TCA is NP-complete.*

5 Conclusion

In this paper, we have introduced and analyzed the Temporal Connectivity Augmentation (TCA) problem and several variants. Our results demonstrate that TCA, in both the strict and non-strict settings, is NP-complete under various constraints, including bounded lifespan and simple graphs.

In addition to these hardness results, we presented polynomial-time algorithms for special cases, such as (1+1)-TCA and TPCA parameterized by the number of pairs. By adapting temporal expansion, we have extended existing techniques to the augmentation setting, and possibly for some weighted temporal problems.

Several promising research directions emerge from our work. A natural extension would involve studying the impact of further structural graph parameters such as those in the work of Enright et al. [13]. Additionally, exploring the problem with restrictions on the underlying graph is also an interesting perspective.

References

1 Eleni C Akrida, Leszek Gąsieniec, George B Mertzios, and Paul G Spirakis. The complexity of optimal design of temporally connected graphs. *Theory of Computing Systems*, 61:907–944, 2017. doi:10.1007/S00224-017-9757-X.

2 Sebastian Angrick, Ben Bals, Tobias Friedrich, Hans Gawendowicz, Niko Hastrich, Nicolas Klodt, Pascal Lenzner, Jonas Schmidt, George Skretas, and Armin Wells. Towards linear spanners in all temporal cliques. *arXiv preprint arXiv:2402.13624*, 2024. doi:10.48550/arXiv.2402.13624.

3 Kyriakos Axiotis and Dimitris Fotakis. On the size and the approximability of minimum temporally connected subgraphs. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 149:1–149:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. `doi:10.4230/LIPICS.ICALP.2016.149`.

4 Sandeep Bhadra and Afonso Ferreira. Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In *Ad-Hoc, Mobile, and Wireless Networks: Second International Conference, ADHOC-NOW2003, Montreal, Canada, October 8-10, 2003. Proceedings 2*, pages 259–270. Springer, 2003. `doi:10.1007/978-3-540-39611-6_23`.

5 Davide Bilò, Gianlorenzo D'Angelo, Luciano Gualà, Stefano Leucci, and Mirko Rossi. Blackout-tolerant temporal spanners. In *International Symposium on Algorithms and Experiments for Wireless Sensor Networks*, pages 31–44. Springer, 2022. `doi:10.1007/978-3-031-22050-0_3`.

6 Mihaela Cardei and Ding-Zhu Du. Improving wireless sensor network lifetime through power aware organization. *Wireless networks*, 11:333–340, 2005. `doi:10.1007/S11276-005-6615-6`.

7 Arnaud Casteigts, Timothée Corsini, and Writika Sarkar. Simple, strict, proper, happy: A study of reachability in temporal graphs. *Theoretical Computer Science*, page 114434, 2024. `doi:10.1016/J.TCS.2024.114434`.

8 Arnaud Casteigts, Joseph G Peters, and Jason Schoeters. Temporal cliques admit sparse spanners. *Journal of Computer and System Sciences*, 121:1–17, 2021. `doi:10.1016/J.JCSS.2021.04.004`.

9 Arnaud Casteigts, Michael Raskin, Malte Renken, and Viktor Zamaraev. Sharp thresholds in random simple temporal graphs. *SIAM Journal on Computing*, 53(2):346–388, 2024. `doi:10.1137/22M1511916`.

10 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999. `doi:10.1006/JAGM.1999.1042`.

11 Esteban Christiann, Eric Sanlaville, and Jason Schoeters. On inefficiently connecting temporal networks. In Arnaud Casteigts and Fabian Kuhn, editors, *3rd Symposium on Algorithmic Foundations of Dynamic Networks, SAND 2024*, volume 292 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. `doi:10.4230/LIPICS.SAND.2024.8`.

12 Jessica Enright, Kitty Meeks, George B Mertzios, and Viktor Zamaraev. Deleting edges to restrict the size of an epidemic in temporal networks. *Journal of Computer and System Sciences*, 119:60–77, 2021. `doi:10.1016/J.JCSS.2021.01.007`.

13 Jessica A. Enright, Samuel D. Hand, Laura Larios-Jones, and Kitty Meeks. Structural parameters for dense temporal graphs. In *49th International Symposium on Mathematical Foundations of Computer Science, MFCS 2024, August 26-30, 2024, Bratislava, Slovakia*, pages 52:1–52:15, 2024. `doi:10.4230/LIPICS.MFCS.2024.52`.

14 Jon Feldman and Matthias Ruhl. The directed steiner network problem is tractable for a constant number of terminals. *SIAM Journal on Computing*, 36(2):543–561, 2006. `doi:10.1137/S0097539704441241`.

15 Aubin Jarry and Zvi Lotker. Connectivity in evolving graph with geometric properties. In *Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pages 24–30, 2004. `doi:10.1145/1022630.1022635`.

16 David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 504–513, 2000. `doi:10.1145/335305.335364`.

17 Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 513–522, 2010. `doi:10.1145/1806689.1806760`.

18 Nathan Michael, Michael M Zavlanos, Vijay Kumar, and George J Pappas. Maintaining connectivity in mobile robot networks. In *Experimental Robotics: The Eleventh International Symposium*, pages 117–126. Springer, 2009.

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

20 Huanhuan Wu, James Cheng, Silu Huang, Yiping Ke, Yi Lu, and Yanyan Xu. Path problems in temporal graphs. *Proceedings of the VLDB Endowment*, 7(9):721–732, 2014. Publisher: VLDB Endowment. doi:10.14778/2732939.2732945.