Theoretical Computer Science 1013 (2024) 114757

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Temporally connected components [☆]

Stefan Balev [a], Eric Sanlaville [a], Jason Schoeters [b,*]

[a] Université Le Havre Normandie, Univ Rouen Normandie, INSA Rouen Normandie, Normandie Univ, LITIS UR 4108, F-76600 Le Havre, France
[b] University of Cambridge, United Kingdom

ARTICLE INFO

Keywords:
Temporal graph theory
NP-complete problems
Polynomial-time reduction
Algorithm
Connected component
Temporal connectivity

ABSTRACT

We discuss a variety of extensions of connected components in temporal graphs, focusing on extensions using connectivity over time through temporal paths (or journeys). Starting with components induced by temporal sources or sinks, we build up to components induced by multiple sources or sinks, and eventually components where all vertices are sources and sinks, i.e. temporally connected components. The cases of bounded components (i.e. defined on time windows), and open or closed components, are also considered. Our contributions mainly include structural results on the number of components, and algorithmic and complexity results of corresponding decision problems. Several new NP-completeness proofs are provided while exploring the boundaries between easy and difficult problems.

1. Introduction

Temporal graphs have become increasingly popular in the literature over the years, and for good reasons. Dynamic settings, whether failure-prone systems or highly mobile entities, which static graphs fail to model, can be modelled naturally with temporal graphs [13,28,31,49]. Many problems in temporal graphs can be solved by extending static graph structures and related problems and algorithms to time dimension, resulting in various more complicated extensions.

Concerning the problems, often these become harder in terms of time and/or space complexity [1,5,40]. Naturally, this leads to results considering specific classes of temporal graphs, often restraining the underlying graph (or footprint), approximation results, and/or fixed-parameter tractability results (or, in the negative case, $W[1]$ or even $W[2]$ hardness results) [15,19,20,23,33,51]. Temporal graph theory has also natural links with gossip theory [4,25,26], and with rainbow structures in edge-coloured static graphs [10,16,37].

We use standard notation and terminology from graph theory [21]. However, temporal graphs can be defined in a number of (more or less) equivalent ways, and as a result exist under a multitude of distinct names, including dynamic graphs, stream graphs, link streams, time-varying graphs and evolving graphs [3,13,22,39]. Usually, the vertex set is constant, and the edges may appear and disappear. We choose the following representation, often referred to as the compact representation: we start from a classical graph (called footprint or underlying graph) and associate to each edge a list of time labels at which the edge is present.

Note that in this largely used model, time is discrete. Discrete temporal graphs may be equivalently represented by a sequence of time indexed static graphs called *snapshots*. It is also possible to represent them by keeping track of the changes between successive

☆ This article belongs to Section A: Algorithms, automata, complexity and games, Edited by Paul Spirakis.

* Corresponding author.
E-mail addresses: stefan.balev@univ-lehavre.fr (S. Balev), eric.sanlaville@univ-lehavre.fr (E. Sanlaville), jason.schoeters.cs@gmail.com (J. Schoeters).
URLs: https://litis.univ-lehavre.fr/~sanlaville/ (E. Sanlaville), https://jschoete.github.io/ (J. Schoeters).

https://doi.org/10.1016/j.tcs.2024.114757
Received 19 September 2023; Received in revised form 17 July 2024; Accepted 24 July 2024

Available online 30 July 2024

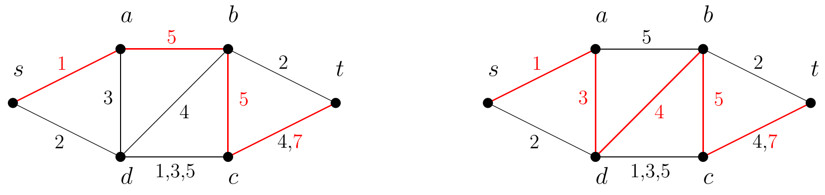

Fig. 1. Example of temporal graph $\mathcal{G}_1$ (presented in the compact representation), with at least two journeys (in red) but no static path from vertex s to vertex t in any of the snapshots. The journey to the right is strict. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

snapshots, which may lead to more compact encoding, especially in the case of low dynamicity (small number of added or removed edges between successive snapshots). It is usually used in the dynamic graph algorithm setting, see [29]. A few works consider continuous models (see for instance [24] for flow problems). However, it is often necessary to discretise the model to obtain algorithmic results.

Journeys and temporal connectivity In static graphs, including snapshots of temporal graphs, vertices s, t are adjacent if they are connected by edge $\{s,t\}$, and edges $\{u,v\},\{v,w\}$ are said to be incident. An (elementary) path is defined as a (non-repeating) sequence of adjacent vertices. If for all pairs of vertices in G there exists a path including both vertices, then G is said to be connected. If G isn't connected, then G can be partitioned into subgraphs such that each subgraph is connected. Such subgraphs, and equivalently the associated vertex sets, are referred to as the connected components of G.

In temporal graphs, one may consider *path-based* connectivity, where the existence of paths between some or all vertices, for each snapshot, is maintained throughout time [46]. However, a natural extension of paths exists in which even though two vertices may admit no path in any of the snapshots, over time they might still be connected thanks to some structures called temporal paths or *journeys*. A journey is a sequence of incident edges (a path in the footprint) with non decreasing time labels. If the time labels are increasing, the journey is called *strict*, see Fig. 1. In this paper, *non-strict* journeys are considered.

The aim of this paper is to present a comprehensive study of the fundamental question *"What is a connected component based on journeys in temporal graphs?"* Some works have appeared over the years offering answers to this question, in one way or another. Here, we wish to group these results, not unlike a survey, and add or complete missing pieces of information relevant to this central question. We choose to consider a well-established hierarchy of temporal connectivity properties proposed by Casteigts et al. in [13], and more recently revisited in [12], as a basis for all sorts of connectivity which may arise or exist in temporal graphs. For some of these properties defined through journeys, we define the corresponding component and present results on computational complexity, algorithms, and the number of components.

Section 2 formalizes the main concepts regarding temporal graphs, journey-based connectivity, and associated components. It then details our main results with regard to previous works, which are summarized in Fig. 2. In Section 3, we state results regarding temporal components in which one vertex connects to/from all others. We show that some of these problems may be solved in polynomial time. Most of these results are new. Then, in Section 4, we present results regarding temporal components in which all vertices connect to/from all others. These results regarding complexity are new or tighten previous results from the literature. Finally, in Section 5, we conclude and discuss some open questions and future works.

2. Related work and main results of the paper

2.1. Main definitions

A temporal graph $\mathcal{G} = (G, \lambda)$ is defined by a static graph $G = (V, E)$ with vertex set $V(G)$ and edge set $E(G) \subseteq V(G) \times V(G)$, and a labelling of the edges $\lambda : E \to 2^{\mathbb{N}} \setminus \emptyset$ determining at which discrete times edges are present. G is commonly called the underlying graph (or *footprint*) of $\mathcal{G}$, sometimes denoted as $G_{\downarrow}(\mathcal{G})$ or simply $G_{\downarrow}$. In this paper, we consider only finite-time temporal graphs, with the largest label, or lifetime, of $\mathcal{G}$ being denoted as $T(\mathcal{G})$. When there is no ambiguity, we denote $V(G)$, $E(G)$ and $T(\mathcal{G})$ as simply V, E and T respectively, and $n = |V|$ and $m = |E|$. We use the notation $G_t(\mathcal{G})$ (or simply G_t) for the static graph corresponding to temporal graph $\mathcal{G}$ at time t, *i.e.* $G_t = (V, E_t \subseteq E)$ with $e \in E_t \iff t \in \lambda(e)$. G_t is called a *snapshot* of $\mathcal{G}$. It is sometimes useful to represent temporal graphs as sequences of snapshots, *e.g.* $\mathcal{G} = (G_1, G_2, ..., G_T)$. Let the restriction of $\mathcal{G}$ on time window $[t_1, t_2]$ for some $t_1 \leq t_2 \leq T$ be the temporal graph denoted $\mathcal{G}_{[t_1,t_2]} = (G_{t_1}, G_{t_1+1}, ..., G_{t_2})$, and $G_{[t_1,t_2]}$ denote its footprint. Finally, let the subgraph of $\mathcal{G}$ induced by vertex set $V' \subseteq V$ be the temporal graph denoted $\mathcal{G}[V'] = (G' = (V', E'), \lambda)$, with $\{u,v\} \in E' \iff u, v \in V'$ and $\{u,v\} \in E$.

We denote a path from u to v as $u \to v$ or, since we consider undirected graphs, $u \leftrightarrow v$. Journeys have been introduced in the previous section as paths of the footprint, each associated with a sequence of non decreasing time labels. In this paper, the journeys may be non-strict, that is successive labels may be equal (or equivalently, a journey may admit several successive edges of the same snapshot). We denote a journey, or the existence of a journey, from u to v, as $u \rightsquigarrow v$. Journeys, as opposed to paths, are neither symmetrical nor transitive, *i.e.* $u \rightsquigarrow v$ and $v \rightsquigarrow w$ does not necessarily imply $u \rightsquigarrow w$. The notation $u \rightsquigarrow v \rightsquigarrow w$ implies that journey $v \rightsquigarrow w$ takes place later in time than journey $u \rightsquigarrow v$ (which also implies $u \rightsquigarrow w$). When a temporal graph $\mathcal{G}$ admits journeys from all vertices to all other vertices, the graph is said to be temporally connected, or $\mathcal{TC}$. We use the notation $u \circlearrowleft v$ for a round-trip journey $u \rightsquigarrow v \rightsquigarrow u$. We note that a journey can be described using temporal logic (see e.g. [44]) as follows: $j(u,v) : (u = v) \vee e(u,v) \vee (\exists w, e(u,w) \wedge j(w,v)) \vee X j(u,v)$, where X denotes the `next` operator.

2.2. Size of a temporal graph encoding

Whether one considers the compact representation, or the sequence of snapshots representation, in both cases the size of a temporal graph is $O(n+mT)$. In specific cases, more appropriate representations exist, such as storing only the differences from one snapshot to the next (added and removed edges), which is specially effective for temporal graphs with few changes between snapshots, or low dynamics. Another possibility is to store for each edge, the sequence of associated presence intervals, effective when the number of these intervals is limited (again this implies a low dynamicity). In this paper however, we do not specifically consider temporal graphs with low dynamicity. Finally, without loss of generality, we assume $m = \Omega(n)$. Indeed, if for some temporal graph $\mathcal{G}$ we have $m < n-1$, *i.e.* the footprint admits multiple connected components V', precomputing allows us to reduce to $O(n)$ temporal graphs $\mathcal{G}' = \mathcal{G}[V']$ where $m' \geq n'-1$.
In this setting, an algorithm on temporal graphs is polynomial if and only if it runs in polynomial time regarding n, m, and T.

2.3. Classification of connectivity properties for temporal graphs

In [13], more recently revisited in [12], Casteigts et al. introduce a hierarchy of temporal properties, giving rise to a number of classes among temporal graphs. We consider in this paper the subset of properties based on the existence of journeys, and that may be used to define different connected components. As stated previously, we also restrict the study to finite-time temporal graphs. The considered properties are classified according to two dimensions. First, existence:

- Property $\mathcal{S}$ (S for "source") indicates there is a vertex s such that $u \rightsquigarrow v$, $\forall v \in V$ (denoted $\mathcal{J}^{1\forall}$ in [13]).
- Property $\mathcal{T}$ (T for "target", or sink) indicates there is a vertex t such that $v \rightsquigarrow t$, $\forall v \in V$ (denoted $\mathcal{J}^{\forall 1}$ in [13]).
- Property $\mathcal{TC}$ as usual indicates $u \rightsquigarrow v$, $\forall u, v \in V$.
- Property $\mathcal{TC}^{\circlearrowleft}$ indicates $u \circlearrowleft v$, $\forall u, v \in V$.

Then, occurrence: journeys may exist during the complete time interval (no superscript). It may also appear several times during this interval. Hence superscript $\mathcal{B}$ will be used if a property is verified during each time window of bounded size Δ. This includes periodic occurrences, that will not be specifically studied here. We also use superscript $\mathcal{D}$ if the source (resp. target) may be different at each time window (dynamic source or target).

For example, $\mathcal{T}$ considers temporal graphs for which a journey exists from any vertex toward some vertex t. $\mathcal{T}^{\mathcal{B}}$ means there are such journeys toward t for each time window of size Δ, where Δ is a given number larger than 1. $\mathcal{T}^{\mathcal{D}}$ considers the previous graphs, for which t may be different for each time window.

We note that in the hierarchy of temporal properties, some properties based on (static) paths and edges are presented. It is possible to naturally define components based on such properties as in [2,46].

2.4. Reachability graph

Besides the footprint, another useful structure concerning temporal graphs is the reachability graph, which represents the reachability (through journeys) of all vertices of the graph. Formally, the reachability graph of a temporal graph $\mathcal{G}$ corresponds to the directed graph $H(\mathcal{G}) = (V, A \subseteq V \times V)$, with $A = \{(v, v') : v \rightsquigarrow v'\}$. Note that this definition introduced by [9] uses the notion of closure in a temporal sense: as stated previously, $u \rightsquigarrow v$ and $v \rightsquigarrow w$ implies $u \rightsquigarrow w$ only if the second journey takes place later than the first one. So $H(\mathcal{G})$ is not, as usual, the reachability graph of some relation, for instance the relation associated to the existence of journeys between vertex couples, but precisely the graph of this relation. When clear from the context, we simply denote $H(\mathcal{G})$ as H. We note that the term "transitive closure" was first used instead of the term "reachability graph", but has now replaced it in the temporal graph community.

For example, the reachability graph of the temporal graph $\mathcal{G}_1$ from Fig. 1 corresponds to a complete bidirectional graph (all arcs in both directions exist) except for arcs (b, s) and (t, s) which are missing, meaning the only journeys missing for $\mathcal{G}_1$ to be temporally connected are from t and b to s.

Constructing the reachability graph can be done in time $O(n(m \log T + n \log n))$ through n calls of an adaptation of Dijkstra's algorithm [48], or in time $O(\mathtt{max}(|E_i|)nT)$ through an online algorithm gradually building the reachability graph snapshot by snapshot [6,7,42]. The latter is more efficient in specific cases when the lifetime $T = O(n)$ and the snapshot density $\mathtt{max}(|E_i|) = o(\log m)$. For simplicity, we will consider the former time complexity throughout this paper, and we will refer to the adaptation of Dijkstra's algorithm as the temporal Dijkstra algorithm.

2.5. From connectivity properties to temporal components

The rest of the paper is organized to define and study, for a given journey-based temporal property $\mathcal{X}$, the analogue of a connected component corresponding to this property, which we will refer to as a $\mathcal{X}$ component in this section.

Definition 1 *($\mathcal{X}$ component).* Given a temporal graph $\mathcal{G}$, a $\mathcal{X}$ component is a maximal subset $V' \subseteq V$ such that $\mathcal{X}$ is respected by V' in $\mathcal{G}$.

Note the maximality requirement for V', which mimics the maximality requirement for static connected components. It is natural to look for largest components also in the temporal case. Definition 1 can be ambiguous concerning the latter part, since technically $\mathcal{X}$ is a property of temporal graphs, not of a set of vertices in a temporal graph. For now, it is sufficient to say that we simply aim to present the difference between $\mathcal{X}$ components and closed $\mathcal{X}$ components, Definition 1 and Definition 2 *resp.* Also, these definitions are formal and clear when presented using concrete properties in Sections 3 and 4.

Using terminology from [9,27,46], we say a $\mathcal{X}$ component V' is closed if journeys between vertices $u, v \in V'$ necessary for property $\mathcal{X}$, do not use vertices from $V \setminus V'$. In other words, to verify if V' is a (potentially non-maximal) closed $\mathcal{X}$ component, it suffices to verify the $\mathcal{X}$ property on $\mathcal{G}[V']$, and this verification does not depend on the rest of $\mathcal{G}$. Going back to the definition of a journey using temporal logic in Section 2.1, a closed $\mathcal{X}$ component thus specifies that w has to be part of V' (and not generally any vertex).

Definition 2 *(Closed $\mathcal{X}$ component).* Given a temporal graph $\mathcal{G}$, a closed $\mathcal{X}$ component is a maximal subset $V' \subseteq V$ such that $\mathcal{X}$ is respected by $\mathcal{G}[V']$.

Open $\mathcal{X}$ components are defined as $\mathcal{X}$ components in which there exists at least one journey necessary for $\mathcal{X}$ which goes outside of the component. Most of the components studied in static graphs are closed (an exception being k-connected components [47] which may be open). In the following, we give results for $\mathcal{X}$ components, and if applicable, also for closed $\mathcal{X}$ components.

For each property $\mathcal{X}$, we start by studying the worst-case number of $\mathcal{X}$ components, *i.e.* the maximum number of $\mathcal{X}$ components which may exist in a given temporal graph. This can be useful for enumeration and partition problems.

For each property $\mathcal{X}$, the corresponding decision problem $\mathcal{X}$ COMPONENT is defined as follows.

Definition 3 *($\mathcal{X}$ COMPONENT decision problem).*
Input: temporal graph $\mathcal{G}$, integer k (and integer Δ if $\mathcal{X}$ is a windowed property).
Question: does $\mathcal{G}$ admit a $\mathcal{X}$ component of size at least k?

Algorithms and complexity results for $\mathcal{X}$ COMPONENT are presented. As stated in Section 1, depending on property $\mathcal{X}$, some results may already exist in the literature. However we additionally determine the boundary (if one exists) between polynomial-time solvability and NP-hardness depending on the lifetime of the graph. To obtain such results, we modify reductions from the literature or use different reductions altogether.

Regarding hardness implications, Bhadra et al. [9] give the following argument for $\mathcal{TC}$ components and closed $\mathcal{TC}$ components, which we generalize for any temporal properties $\mathcal{X}_1$, $\mathcal{X}_2$. $\mathcal{X}_1$ components are a special case of $\mathcal{X}_2$ components, but the NP-hardness of $\mathcal{X}_1$ COMPONENT does not directly imply that $\mathcal{X}_2$ COMPONENT is NP-hard as well. This is because a possible polynomial time algorithm for $\mathcal{X}_2$ COMPONENT need only answer the decision problem and not identify the components of size at least k, thus potentially making it difficult to verify if at least one such a component is a $\mathcal{X}_1$ component. Also, the same temporal graph may contain both a $\mathcal{X}_1$ component (of indeterminate size) and a $\mathcal{X}_2$ component of size k, so the decision problem for the latter would always return "yes", ignoring the presence or absence of a $\mathcal{X}_1$ component of size k, thereby leaving its decision problem unsolved. Of course, the other way around, since $\mathcal{X}_1$ components are a special case of $\mathcal{X}_2$ components, if $\mathcal{X}_2$ COMPONENT is NP-hard, then $\mathcal{X}_1$ COMPONENT is not necessarily NP-hard either. Since hardness results do not transfer one way or the other, note that this implies that neither do results on polynomial-time solving. This is a major difference with the study of connectivity at the graph level.

Regarding hardness proofs, *i.e.* reductions, we use the following trick from [9]: if all $\mathcal{X}$ components in the transformed instance are closed, then the reduction works for both $\mathcal{X}$ COMPONENT and CLOSED $\mathcal{X}$ COMPONENT.

2.6. Summary of results

A concise summary of this paper results is given in the following table, where we precise for each $\mathcal{X}$ component the worst-case number of components (parameterised by the number of vertices n of the temporal graph), or the presented bounds on this number using notation [`lower bound`, `upper bound`]. We also present the algorithmic complexity of finding a solution, if any, and the complexity class of the corresponding decision problem (P referring to polynomial-time solvable, NPC to NP-complete). For $\mathcal{X}$ components defined through the $\mathcal{TC}$ property (lower part of the table) we give the constant values of T (and Δ when applicable) for which the corresponding decision problem is NP-complete. Results for $\mathcal{T}$ properties are not presented. As we shall proof in section 3, the results for $\mathcal{S}$ properties easily transfer to them. Our study shows that NP-hardness basically comes from a non polynomial number of potential vertex subsets to check.

Temporal graphs with lifetime $T = 1$ trivially admit polynomial-time algorithms for all these problems (since they reduce to finding connected components in the static graph), so they are not presented. We also prove $\mathcal{TC}^{\circlearrowleft}$ COMPONENT and CLOSED $\mathcal{TC}^{\circlearrowleft}$ COMPONENT can be solved in polynomial time for temporal graphs with lifetime $T = 2$. Note that all these results are proven in this paper, although some parts (about $\mathcal{TC}$ and alike) were first presented in related works: Casteigts [12], adapting the work of [41], provided first results on the number of $\mathcal{TC}$ components. Bhadra et al. [8] proved that $\mathcal{TC}$ COMPONENT is NP-complete for $T \geq 4$, and provided an algorithm studied by Nicosia et al. [43]. Very recently, Costa et al. [18] presented new parameterized complexity results for $\mathcal{TC}$ COMPONENT. Gómez-Calzado et al. [27] studied the existence of $\mathcal{TC}^{\mathcal{B}}$ Components, and Huyghues-Despointes et al. [32] their maximum size.

Component	Worst-case Number	Algorithmic complexity	Class
$\mathcal{S}$	n ($\frac{n}{2}$ if no isolated vertices)	$O(n(m\log T + n\log n))$	P
$\mathcal{S}^{\mathcal{B}}$	n	$O((T-\Delta)n(m\log\Delta + n\log n))$	P
Closed $\mathcal{S}^{\mathcal{B}}$	n	$O(n(T-\Delta)(n-k)(m\log\Delta + n\log n))$	P
$\mathcal{S}^{\mathcal{D}}$	$[n, \min(2^n, n^{T-\Delta+1})]$	$O(n^{\min(k,T-\Delta+2)}(T-\Delta)k(m\log\Delta + n\log n))$	NPC
Closed $\mathcal{S}^{\mathcal{D}}$	$[n, \min(2^n, n^{T-\Delta+1})]$	$O(\min(2^n, n^{T-\Delta+1})(T-\Delta)n(m\log\Delta + n\log n))$	NPC
$\mathcal{TC}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O(nm\log T + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$	NPC ($T > 1$)
Closed $\mathcal{TC}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O(2^n n(m\log T + n\log n))$	NPC ($T > 1$)
$\mathcal{TC}^{\mathcal{B}}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O((T-\Delta)n(m\log\Delta + n\log n) + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$	NPC ($T\&\Delta > 1$)
Closed $\mathcal{TC}^{\mathcal{B}}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O(2^n(T-\Delta)n(m\log\Delta + n\log n))$	NPC ($T\&\Delta > 1$)
$\mathcal{TC}^{\circlearrowleft}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O(n^2(m\log T + n\log n) + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$	NPC ($T > 2$)
Closed $\mathcal{TC}^{\circlearrowleft}$	$[2^{0.52\sqrt{n}}, 2^{0.53n}]$	$O(2^n(n^2(m\log T + n\log n))$	NPC ($T > 2$)

Fig. 2. Table of results concerning the type of connectivity and worst-case number, algorithmic complexity, and complexity class.

2.7. *Other problems related to connectivity*

Of course there are other questions linked to connectivity, but they are out of scope of this paper. The interested reader might look at [20,48], among many others, for shortest paths, [15] for spanners and spanning trees, [33] for temporal exploration.

3. One to all / one from all ($\mathcal{S}$, $\mathcal{S}^{\mathcal{B}}$, $\mathcal{S}^{\mathcal{D}}$, $\mathcal{T}$, $\mathcal{T}^{\mathcal{B}}$, $\mathcal{T}^{\mathcal{D}}$)

A temporal source is a vertex u such that all other vertices v in the temporal graph admit a journey $u \rightsquigarrow v$. The concept is mainly useful to model a network over which one agent, say the leader, can control the entire network, diffusing information, messages or influence throughout the network using peer-to-peer broadcasting [17,34,50]. In this section, we prove that in most cases, the problem of finding, and even enumerating the connected components is polynomial. The exception is the case of dynamic sources ($\mathcal{D}$ superscript), where the number of solutions can be non polynomial.

Definition 4 *($\mathcal{S}$ component).* An $\mathcal{S}$ component of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\exists u \in V'$, $\forall v \in V'$, $u \rightsquigarrow v$ in $\mathcal{G}$.

Adding the natural constraint of time windows has mainly two advantages, first a time bound after which one is ensured all other vertices can be reached from the source, and second the possibility for the source to reach other vertices multiple times over the lifetime of the network.

Definition 5 *($\mathcal{S}^{\mathcal{B}}$ component).* An $\mathcal{S}^{\mathcal{B}}$ component of duration Δ of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\exists u \in V'$, $\forall t \le T - \Delta + 1$, $\forall v \in V'$, $u \rightsquigarrow v$ in $\mathcal{G}_{[t,t+\Delta-1]}$.

A natural relaxation of the latter allows for any vertex to be the source in a time window, not necessarily the same vertex for all time windows. In other words, the source may be dynamic and change over time.

Definition 6 *($\mathcal{S}^{\mathcal{D}}$ component).* An $\mathcal{S}^{\mathcal{D}}$ component of duration Δ of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\forall t \le T - \Delta + 1$, $\exists u \in V'$, $\forall v \in V'$, $u \rightsquigarrow v$ in $\mathcal{G}_{[t,t+\Delta-1]}$.

Closely related to source vertices are sink vertices, where other vertices are able to reach such a vertex, leading to properties $\mathcal{T}$, $\mathcal{T}^{\mathcal{B}}$, and $\mathcal{T}^{\mathcal{D}}$. These problems are often studied independently as they have their own specific applications such as collecting data to analyse a network, and broadcasting information to control a network (e.g. [30, 33, 47]). However, for intents and purposes in this paper, these problems can be treated in an equivalent manner by reversing the timeline as presented below (also used recently in [11]).

Lemma 1. *$\mathcal{T}$ (resp. $\mathcal{S}$, $\mathcal{T}^{\mathcal{B}}$, $\mathcal{S}^{\mathcal{B}}$, $\mathcal{T}^{\mathcal{D}}$, $\mathcal{S}^{\mathcal{D}}$) components in temporal graph $\mathcal{G} = (G, \lambda)$ of lifetime T correspond to $\mathcal{S}$ (resp. $\mathcal{T}$, $\mathcal{S}^{\mathcal{B}}$, $\mathcal{T}^{\mathcal{B}}$, $\mathcal{S}^{\mathcal{D}}$, $\mathcal{T}^{\mathcal{D}}$) components in temporal graph $\mathcal{G}' = (G, \lambda')$ of lifetime T, where $\forall e \in E(G), \lambda'(e) = \bigcup_{\ell \in \lambda(e)} T - \ell$.*

Proof. Any journey in $\mathcal{G}$ from vertex u to v is reversed in $\mathcal{G}'$. Thus, a vertex able to reach all vertices in $\mathcal{G}$ (or in some $\mathcal{G}_{[t,t+\Delta-1]}$) can be reached by all vertices in $\mathcal{G}'$ (or in some $\mathcal{G}'_{[t',t'+\Delta-1]}$) and vice versa. ◀

Lemma 1 allows us to focus only on $\mathcal{S}$ components, since any structural result transfers to $\mathcal{T}$ components, and algorithmic results transfer too, with a polynomial overhead of $O(mT)$ to build the reversed graph (but they can be adapted easily without any overhead). The same holds for $\mathcal{S}^{\mathcal{B}}$ components and $\mathcal{T}^{\mathcal{B}}$ components, as well as for $\mathcal{S}^{\mathcal{D}}$ components and $\mathcal{T}^{\mathcal{D}}$ components.

3.1. $\mathcal{S}$ components

Lemma 2. *$\mathcal{S}$ components are necessarily closed.*

Proof. In an $\mathcal{S}$ component, all vertices on a journey from the source to some other vertex also admit a journey from the source, and are thus by maximality included in the component as well. ◀

Lemma 3. *A vertex can only be a source for one $\mathcal{S}$ component.*

Proof. If a vertex was a source for two distinct $\mathcal{S}$ components, then at least one wouldn't be maximal since their union would result in a larger $\mathcal{S}$ component. ◀

Observation 1. *An isolated vertex in the footprint is an $\mathcal{S}$ component.*

Theorem 1. *The worst-case number of $\mathcal{S}$ components is n.*

Proof. Lemma 3 implies that at most n $\mathcal{S}$ components can exist, which by Observation 1 is tight when considering the trivial case of the empty temporal graph, *i.e.* a temporal graph composed of empty graph snapshots $G_i = (V, \emptyset)$. ◀

Concerning the worst-case number of $\mathcal{S}$ components, we establish a relation between $\mathcal{S}$ components and locally minimum label edge sets, the latter being defined as maximal connected edge sets such that all edges share some common label and such that no edges incident to this set exist with smaller labels. A similar concept called locally earliest edges was used recently in [38].

Lemma 4. *The number of $\mathcal{S}$ components is at most the number of locally minimum edge sets in a temporal graph without isolated vertices.*

Proof. For any locally minimum label edge set $E' = \{\{u_1, u_2\}, \ldots \{u_j, u_k\}\}$, observe that $\mathcal{S}$ components S_{u_i} with source $u_i \in V(E')$ are identical (in terms of their vertex set), since all vertices u_i can reach all vertices reachable by all other vertices of E' through the locally minimum label edge set. Hence, such vertices u_i of a locally minimum label edge set produce altogether at most one $\mathcal{S}$ component. Concerning vertices which are not part of a locally minimum label edge set, say vertex v, observe that since v is not part of a locally minimum label edge set, this means that its incident edge with the smallest label, say $\{v, x\}$, must have some incident edge $\{x, y\}$ with an even smaller label (or $\{v, x\}$ is part of a connected edge set of the same label which has such an incident edge). Now, either $\{x, y\}$ is part of a locally minimum label edge set or it has again some incident edge with a smaller label (or again its connected edge set of the same label does). Continue this process until having found a locally minimum label edge set, say $E' = \{\{u_1, u_2\}, \ldots \{u_j, u_k\}\}$. Note that any u_i can reach v at the latter's earliest incident edge's label, implying any vertex v can reach, can be reached by u_i as well, through v. $\mathcal{S}$ component S_v with source v thus has to be contained (in terms of their vertex set) within the larger $\mathcal{S}$ component S_{u_i} with source u_i, since u_i can reach all vertices v can reach. Hence, vertices v which are not part of a locally minimum label edge set cannot be the source of a $\mathcal{S}$ component, since it would not be maximal. This combined with locally minimum label edge sets producing at most one $\mathcal{S}$ component each, effectively bounds the number of $\mathcal{S}$ components by the number of locally minimum label edge sets. ◀

Note that there is not necessarily equality for Lemma 4, for example when one component induced by a locally minimum label edge set is included in another, such as shown in Fig. 3.

Lemma 5. *At most $\frac{n}{2}$ locally minimum label edge sets may exist in any temporal graph, which is tight.*

Proof. Since locally minimum label edge sets cannot be incident by definition, and cover at least two vertices each, at most $\frac{n}{2}$ locally minimum label edge sets exist in any graph. This is shown tight as follows. Take a complete graph on n vertices $K_n = (V, E)$ without any labels. Now, take a maximum matching in K_n and assign small labels to it, and to the rest of the edges, either assign large labels or remove these edges. The result is a temporal graph with exactly $\frac{n}{2}$ locally minimum label edge sets, corresponding to the $\frac{n}{2}$ edges of the maximum matching. ◀

Theorem 2. *The worst-case number of $\mathcal{S}$ components is $\frac{n}{2}$ in temporal graphs without isolated vertices.*

Proof. Fig. 4 shows a graph family which admits exactly $\frac{n}{2}$ $\mathcal{S}$ components, which by Lemma 4 and Lemma 5 finishes the proof. ◀

Theorem 3. *$\mathcal{S}$ COMPONENT is solvable in polynomial time $O(n(m \log T + n \log n))$.*

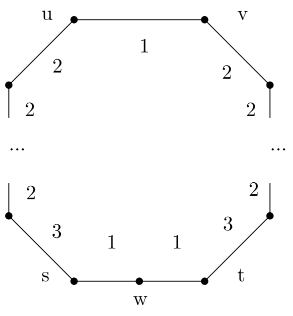

Fig. 3. Example graph family for which Lemma 4 does not induce an equality between the number of $\mathcal{S}$ components (only one exists composed of all vertices with source either u or v) and the number of locally minimum edge sets (two such sets exist: $\{u,v\}$ and $\{\{s,w\},\{w,t\}\}$).

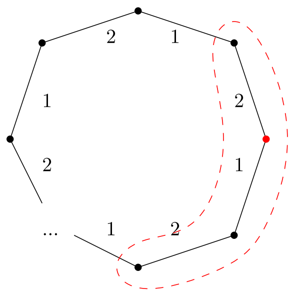

Fig. 4. A graph family with $\frac{n}{2}$ $\mathcal{S}$ components. A component with its corresponding source is marked in red.

Proof. We propose the following algorithm for $\mathcal{S}$ COMPONENT. First, construct the reachability graph H of the given temporal graph $\mathcal{G}$, then check if $k \le \Delta^+(H) + 1$, where $\Delta^+(H)$ denotes the maximum out degree of H. The described algorithm has a time complexity of $O(n(m\log T + n\log n))$ to construct the reachability graph H, and an additional $O(n^2)$ to compute the maximum out degree $\Delta^+(H)$. ◀

Theorem 4. *For a given temporal graph, all $\mathcal{S}$ components can be enumerated in time $O(nm\log T + n^3)$.*

Proof. Start by computing the reachability graph H, taking time $O(n(m\log T + n\log n))$. Any $\mathcal{S}$ component is a subset composed of some vertex v (the source) with all its out-neighbours in H. Now, some of these subsets may be included into others, meaning they may be non-maximal. To remove non-maximal subsets, start by sorting the vertices in each subset, taking time $O(n^2\log n)$. Then, for each pair of subsets, of which there are $O(n^2)$, check whether one is a subset of the other, and remove it if so, which can be done in time $O(n)$ after sorting. All in all, enumeration of $\mathcal{S}$ components can be done in time $O(n(m\log T + n\log n) + n^2\log n + n^3) = O(nm\log T + n^3)$. ◀

3.2. $\mathcal{S}^{\mathcal{B}}$ components

Observation 2. *Contrary to $\mathcal{S}$ components, $\mathcal{S}^{\mathcal{B}}$ components can be open, as shown in Fig. 5.*

Lemma 6. *A vertex can only be a source for one $\mathcal{S}^{\mathcal{B}}$ component.*

Proof. If a vertex was a source for two distinct $\mathcal{S}^{\mathcal{B}}$ components, then at least one wouldn't be maximal since their union would result in a larger $\mathcal{S}^{\mathcal{B}}$ component. ◀

The above results are true also for closed components. We first focus on the general case: the components might be open.

3.2.1. General case

Theorem 5. *The worst-case number of $\mathcal{S}^{\mathcal{B}}$ components is n.*

Proof. Lemma 6 implies at most n $\mathcal{S}^{\mathcal{B}}$ components may exist, which is tight for example for the empty temporal graph. ◀

In fact, this is tight even for much denser graphs; a temporal graph with one window of size Δ containing only empty snapshots is sufficient to obtain n $\mathcal{S}^{\mathcal{B}}$ components of size 1, meaning even a temporal graph with a complete graph as footprint may attain n $\mathcal{S}^{\mathcal{B}}$ components.

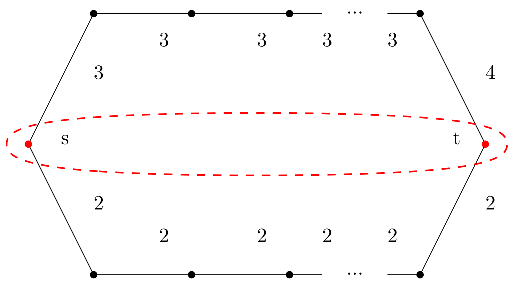

Fig. 5. A graph family admitting an open $\mathcal{S}^{\mathcal{B}}$ component for $\Delta = 2$ (in dashed and red with vertex s as source) and an open $\mathcal{T}^{\mathcal{B}}$ component for $\Delta = 2$ (in dashed and red with vertex t as sink).

Theorem 6. *$\mathcal{S}^{\mathcal{B}}$ COMPONENT is solvable in polynomial time $O((T - \Delta)n(m \log \Delta + n \log n))$.*

Proof. We propose the following algorithm for $\mathcal{S}^{\mathcal{B}}$ COMPONENT. Start by enumerating the $\mathcal{S}$ components of the windowed temporal graph $\mathcal{G}_{[1,\Delta]}$, which by Theorem 4 can be done in time $O(nm \log \Delta + n^3)$. However, we are not interested in maximal $\mathcal{S}$ components, so the added $O(n^3)$ can be removed. Thus, we enumerate $\mathcal{S}$ components, some of which may not be maximal, in time $O(n(m \log \Delta + n \log n))$. Keep only those of size at least k. These are candidate components for $\mathcal{S}^{\mathcal{B}}$ components. Let us denote these candidate components as S_s by their unique corresponding source vertex s (Lemma 3). Then, for each windowed temporal graph $\mathcal{G}_{[t,t+\Delta-1]}$, of which there are $O(T - \Delta)$, obtain their $\mathcal{S}$ components in the same manner, again in time $O(n(m \log \Delta + n \log n))$. Denote these $\mathcal{S}$ components as $S_s[t, t + \Delta - 1]$. For each candidate component S_s, of which there are $O(n)$, update S_s by setting it to the intersection of S_s and $S_s[t, t + \Delta - 1]$ which is doable in time $O(n \log n)$. Remove S_s from the candidate components if its size is less than k. If no candidate components remain at some point, the instance is negative, otherwise it is positive. All in all, this gives a total time complexity of $O(n(m \log \Delta + n \log n) + (T - \Delta)n(m \log \Delta + n \log n + n(n \log n))) = O((T - \Delta)n(m \log \Delta + n \log n))$. ◀

Theorem 7. *For a given temporal graph, all $\mathcal{S}^{\mathcal{B}}$ components can be enumerated in time $O((T - \Delta)n(m \log T + n \log n) + n^3)$.*

Proof. Run the algorithm from Theorem 6 (except for the removal of candidates of size less than k) and at the end add the verification and removal of non-maximal $\mathcal{S}^{\mathcal{B}}$ components, doable as in Theorem 4 in time $O(n^3)$, for a total of $O((T - \Delta)n(m \log T + n \log n) + n^3)$. ◀

3.2.2. *Closed $\mathcal{S}^{\mathcal{B}}$ components*

Theorem 8. *The worst-case number of closed $\mathcal{S}^{\mathcal{B}}$ components is n.*

Proof. Theorem 5 directly adapts for closed $\mathcal{S}^{\mathcal{B}}$ components, since n $\mathcal{S}^{\mathcal{B}}$ components of size 1 are necessarily closed. ◀

Lemma 7. *Verifying if a vertex subset S_s of source s and of size k is a (potentially non-maximal) closed $\mathcal{S}^{\mathcal{B}}$ component takes time $O((T - \Delta)(\min(m, k^2) \log \Delta + k \log k))$.*

Proof. For each windowed temporal subgraph $\mathcal{G}[S_s]_{[t,t+\Delta-1]}$, of which there are $O(T - \Delta)$, apply the temporal Dijkstra algorithm from vertex s to obtain its reachability which is done in time $O(\min(m, k^2) \log \Delta + k \log k)$. If for all windows, its reachability covers the whole vertex set S_s, then S_s is a solution, else it is not. ◀

Theorem 9. *CLOSED $\mathcal{S}^{\mathcal{B}}$ COMPONENT is solvable in polynomial time $O(n(n - k)(T - \Delta)(m \log \Delta + n \log n))$.*

Proof. We propose the following algorithm for CLOSED $\mathcal{S}^{\mathcal{B}}$ COMPONENT. Run the algorithm for $\mathcal{S}^{\mathcal{B}}$ COMPONENT. At the end of the algorithm, the set of candidates contains all $\mathcal{S}^{\mathcal{B}}$ components S_s of size at least k. For all S_s, test if it is closed, *i.e.* if for all $t \leq T - \Delta + 1$, S_s is an $\mathcal{S}$ component of temporal graph $\mathcal{G}[S_s]_{[t,t+\Delta-1]}$. If some component is, then the instance is positive. If however S_s is open, meaning at some time $t \leq T - \Delta + 1$, vertex s cannot reach all vertices in $\mathcal{G}[S_s]_{[t,t+\Delta-1]}$, then S_s can be shrunk down to contain only the reachable vertices in $\mathcal{G}[S_s]_{[t,t+\Delta-1]}$. This shrunken down S_s can now be tested again: if it is of size at least k and closed, then the instance is positive, and if not we can again shrink S_s *etc.* If no closed $\mathcal{S}^{\mathcal{B}}$ component remains of suitable size, then the instance is negative. The algorithm for $\mathcal{S}^{\mathcal{B}}$ COMPONENT runs in time $O((T - \Delta)n(m \log \Delta + n \log n))$, after which $O(n)$ candidate components are tested whether they contain a large enough closed $\mathcal{S}^{\mathcal{B}}$ component. For each of these, testing if they are closed takes time $O((T - \Delta)(m \log \Delta + n \log n))$ by Lemma 7. This process is repeated in case the component is not closed and a shrunken component has to be tested which again isn't closed but contains another shrunken component to be tested *etc.* This can happen at most $n - k$ times, since the size of a component is at most n and we're not interested in components of size less than k. Hence, all together, we obtain a complexity of $O(n(T - \Delta)(m \log \Delta + n \log n) + n(n - k)(T - \Delta)(m \log \Delta + n \log n)) = O(n(n - k)(T - \Delta)(m \log \Delta + n \log n))$. ◀

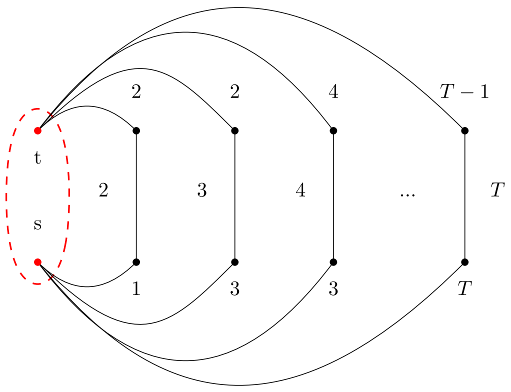

Fig. 6. A graph family admitting an open $\mathcal{S}^{\mathcal{D}}$ component for $\Delta = 2$, with T odd (in dashed and red with vertex s as source for windows $[\texttt{odd}, \texttt{even}]$ and t as source for windows $[\texttt{even}, \texttt{odd}]$).

3.3. $\mathcal{S}^{\mathcal{D}}$ components

Some results from $\mathcal{S}^{\mathcal{B}}$ components transfer for $\mathcal{S}^{\mathcal{D}}$ components, or give bounds, since $\mathcal{S}^{\mathcal{B}}$ components are $\mathcal{S}^{\mathcal{D}}$ components in which the source for every window is the same vertex.

Observation 3. *Even $\mathcal{S}^{\mathcal{D}}$ components where sources change between successive windows can be open, such as shown in Fig. 6.*

Lemma 8. *An ordered set of vertices (one source per window possibly with repetitions) corresponds to sources of at most one $\mathcal{S}^{\mathcal{D}}$ component.*

Proof. If such a set of vertices were sources for two distinct $\mathcal{S}^{\mathcal{D}}$ components, then at least one wouldn't be maximal since their union would result in a larger $\mathcal{S}^{\mathcal{D}}$ component. ◀

3.3.1. General case

Theorem 10. *The worst-case number of $\mathcal{S}^{\mathcal{D}}$ components is at least n, and at most* $\min(2^n, n^{T-\Delta+1})$.

Proof. The lower bound comes from Theorem 8, with an example being the empty temporal graph. For the upper bounds, since $\mathcal{S}^{\mathcal{D}}$ components are subsets of vertices which may intersect, at most 2^n $\mathcal{S}^{\mathcal{D}}$ components may exist. By Lemma 8, representing components through their sources, *i.e.* ordered sets of vertices (one source per window), another bound of $n^{T-\Delta+1}$ is found. ◀

Lemma 9. *Verifying if a vertex subset V' of size k is a (potentially non-maximal) $\mathcal{S}^{\mathcal{D}}$ component takes time $O((T-\Delta)k(m\log\Delta + n\log n))$.*

Proof. For each window $[t, t+\Delta-1]$, of which there are $T-\Delta+1$, one constructs a partial reachability graph H' of $\mathcal{G}_{[t,t+\Delta-1]}$ in which only reachability for the vertices V' is computed. This takes time $O(k(m\log\Delta + n\log n))$. For each induced subgraph $H'[V']$, checking if a vertex with out degree $k-1$ exists takes time $O(k^2)$. In total, this takes time $O((T-\Delta)(k(m\log\Delta + n\log n) + k^2) = O((T-\Delta)k(m\log\Delta + n\log n))$. ◀

Theorem 11. *$\mathcal{S}^{\mathcal{D}}$ COMPONENT is solvable in time $O(n^{\min(k,T-\Delta+2)}(T-\Delta)k(m\log\Delta + n\log n))$.*

Proof. We present two algorithms, the first being the brute force algorithm that tests all possible subsets of vertices of size k, the second a generalization of the algorithm for $\mathcal{S}^{\mathcal{B}}$ COMPONENT. Let us start by analysing the brute force algorithm. For all possible subsets of vertices, of which there exist $\binom{n}{k} = O(n^k)$, we verify if it is a suitable solution. By Lemma 9, one verification takes time $O((T-\Delta)k(m\log\Delta + n\log n))$, meaning the brute force algorithm runs in time $O(n^k(T-\Delta)k(m\log\Delta + n\log n))$. Concerning the generalization of the algorithm for $\mathcal{S}^{\mathcal{B}}$ COMPONENT, only one change needs to be made. Instead of shrinking candidate component S_s at every window by taking the union with $S_s[t, t+\Delta-1]$, we will need to instead shrink the candidate component with all $S_v[t, t+\Delta-1]$ for all $v \in S_s$ (which could all correspond to a source for that time window), splitting a candidate component up into potentially $O(n)$ distinct components every time window. A result of this is that at every time window $[t, t+\Delta-1]$, not $O(n)$ candidate components S_s exist from previous windows, but at most $n^t = O(n^{T-\Delta+1})$ candidate components $S_{(s,t,u,\ldots,v)}$. Analysing the complexity gives $O(n^{T-\Delta+1}n(T-\Delta)(m\log\Delta + n\log n))$. ◀

Theorem 12. *$\mathcal{S}^{\mathcal{D}}$ COMPONENT is NP-complete, even for window size $\Delta = 2$.*

Proof. Lemma 9 proves $\mathcal{S}^{\mathcal{D}}$ COMPONENT is in NP. To prove NP-hardness, we give a polynomial-time reduction from CLIQUE to $\mathcal{S}^{\mathcal{D}}$ COMPONENT, *i.e.* we show how to transform any instance of CLIQUE to an instance of $\mathcal{S}^{\mathcal{D}}$ COMPONENT such that the instance of CLIQUE is positive if and only if the instance of $\mathcal{S}^{\mathcal{D}}$ COMPONENT is positive.

Given an instance of CLIQUE, being a graph G_1 of n_1 vertices and m_1 edges and an integer k_1, we describe the following polynomial-time transformation to an instance of $\mathcal{S}^{\mathcal{D}}$ COMPONENT, being a temporal graph $\mathcal{G}_2 = (G_2, \lambda_2)$ of lifetime T_2 and integers Δ_2 and k_2. Set $T_2 = 3n_1 - 1$, $\Delta_2 = 2$ and $k_2 = m_1 + n_1 + k_1$. Let us construct the footprint G_2, initially the empty graph. Let $V(G_1) = \{v_1, v_2, ..., v_{n_1}\}$. For each vertex $v_i \in V(G_1)$, add vertices v_i and $v_{i,i}$ with edge $\{v_i, v_{i,i}\}$ to G_2. We refer to vertices v_i in G_2 as original vertices, and to vertices $v_{i,i}$ as satellite vertices. Then, for each edge $\{v_i, v_j\} \in E(G_1)$, add vertex $v_{i,j} = v_{j,i}$ and edges $\{v_i, v_{i,j}\}$, $\{v_{i,j}, v_j\}$ to G_2. These edges will allow for v_i to reach v_j and/or vice versa, depending on the labelling. We refer to vertices $v_{i,j}$ as intermediary vertices. Finally, $\forall v_{i,j}, v_{h,k} \in G_2$, add edges $\{v_{i,j}, v_{h,k}\}$ (essentially creating a clique of all intermediary and satellite vertices, of size $m_1 + n_1$). Let us refer to these last edges as the background edges. This concludes the construction of G_2. Note that $n_2 = m_1 + 2n_1$.
Concerning λ_2 (see also Fig. 7), for each vertex $v_a \in V(G_1)$ with $1 \le a \le n_1$, we create a temporal graph $\mathcal{G}_{v_a}$ Construct each $\mathcal{G}_{v_a}$ by placing label $3a - 1$ on all edges $\{v_i, v_{i,a}\}$ and on the background edges, and label $3a - 2$ on all other edges. Note that the intermediary vertices allow for v_i and v_j to reach each other, except for temporal graphs $\mathcal{G}_{v_i}$ in which only v_i can reach v_j, and vice versa for $\mathcal{G}_{v_j}$. Now, let $\mathcal{G}_2$ be the union of all $\mathcal{G}_{v_a}$, for all $1 \le a \le n_1$, and add labels $3a$ on all edges, for all $1 \le a \le n_1 - 1$. This concludes the polynomial-time transformation.

Now let's show that the instance of CLIQUE, being (G_1, k_1), is positive if and only if the instance of $\mathcal{S}^{\mathcal{D}}$ COMPONENT, being $(\mathcal{G}_2, \Delta_2 = 2, k_2)$, is positive.

(G_1, k_1) **is positive** $\implies$ $(\mathcal{G}_2, \Delta_2, k_2)$ **is positive:**
Suppose that a clique exists in G_1 of size k_1, composed of vertices $V' = v_h,\ v_{h+1}, ...v_{h+k_1}$. Note that in all windows of size $\Delta_2 = 2$ in $\mathcal{G}_2$, either the window contains $3a$ for some a and thus all vertices can reach each other in this window using only edges with label $3a$, or the window does not contain $3a$ in which case it must contain $3a - 1$, meaning $m_1 + n_1$ vertices can reach each other by using the background edges with label $3a - 1$. In either case, this implies a $\mathcal{S}^{\mathcal{D}}$ component of size at least $m_1 + n_1$ must exist in $\mathcal{G}_2$. To prove a larger $\mathcal{S}^{\mathcal{D}}$ component exists, of size $k_2 = m_1 + n_1 + k_1$, we can ignore windows containing $3a$, since in these windows a $\mathcal{S}$ component of size $n_2 \ge k_2$ exists and can thus be shrunk to adapt to any $\mathcal{S}^{\mathcal{D}}$ component suitable for the other windows. The remaining windows correspond exactly to the temporal graphs $\mathcal{G}_{v_i}$ for all $1 \le i \le n_1$. Now, all edges $\{v_i, v_j\}$ composing our clique in G_1 exist by definition as edges $\{v_i, v_{i,j}\}$ and $\{v_{i,j}, v_j\}$, with either labels 1 and 1 *resp.* or 1 and 2 *resp.*, or 2 and 1 *resp.* in temporal graphs $\mathcal{G}_{v_a}$. Let us iterate over these temporal graphs (with variable b), in which two cases are distinguished:

- If the edges of the clique in G_1 are all transformed in edges with label 1 in the temporal graph $\mathcal{G}_{v_b}$, then vertex v_h is a source able to reach V' in $\mathcal{G}_{v_b}$, for a total of at least $m_1 + n_1 + k_1 = k_2$ vertices.
- If some edges of the clique in G_1 are transformed into edges with label 2 in the temporal graph $\mathcal{G}_{v_b}$, then v_b must be a part of the clique in G_1, since by definition only adjacent edges to v_b are transformed into edges with label 2 in $\mathcal{G}_{v_b}$ (outside of background edges). Also, v_b (and only v_b amongst the original vertices) corresponds to a source able to reach V' in $\mathcal{G}_{v_b}$, for a total of at least $m_1 + n_1 + k_1 = k_2$ vertices.

In both cases, and thus for all $\mathcal{G}_{v_a}$, a $\mathcal{S}$ component exists containing the aforementioned $m_1 + n_1$ vertices, as well as the k_1 vertices from the clique in G_1. The result of a $\mathcal{S}^{\mathcal{D}}$ component of size $m_1 + n_1 + k_1 = k_2$ existing in $\mathcal{G}_2$ follows directly.

$(\mathcal{G}_2, \Delta_2, k_2)$ **is positive** $\implies$ (G_1, k_1) **is positive:**
Again, observe that any $\mathcal{S}^{\mathcal{D}}$ component in $\mathcal{G}_2$ must contain the $m_1 + n_1$ vertices which the background edges cover. Let us look at the other $k_2 - (m_1 + n_1) = k_1$ vertices. These vertices must be v_i for some i (since vertices $v_{i,j}$ are already contained through the background edges), meaning they correspond directly to vertices v_i of G_1. Let's denote these vertices as V'. Consider temporal graph $\mathcal{G}_{v_b}$ for some v_b, and observe that any vertex v_i can, by construction, only reach other vertices v_j by traversing intermediary vertices $v_{i,j}$. Also, at most one original vertex v_j can be reached by a journey starting at a vertex v_i. This implies all vertices in V' must be linked pairwise through intermediary vertices in $\mathcal{G}_{v_b}$. By construction, intermediary vertices between vertices V' can only exist in $\mathcal{G}_{v_b}$ if the corresponding edges exist between V' in G_1. Thus a clique of size $|V'| = k_1$ must exist in G_1. ◀

3.3.2. *Closed $\mathcal{S}^{\mathcal{D}}$ components*

Theorem 13. *The worst-case number of closed $\mathcal{S}^{\mathcal{D}}$ components is at least n, and at most* $\min(2^n, n^{T-\Delta+1})$.

Proof. The lower bound from Theorem 10 works as $\mathcal{S}^{\mathcal{D}}$ components of size 1 are closed. The upper bounds transfer directly from Theorem 10. ◀

Lemma 10. *Verifying if a vertex subset V' of size k is a (potentially non-maximal) closed $\mathcal{S}^{\mathcal{D}}$ component takes time* $O((T - \Delta)k(\min(m, k^2)\log\Delta + k\log k))$.

Proof. For each window $[t, t + \Delta - 1]$, of which there are $T - \Delta$, constructing the reachability graph of $\mathcal{G}[V']_{[t,t+\Delta-1]}$ takes time $O(k(\min(m, k^2)\log\Delta + k\log k))$. In each reachability graph, checking if a vertex with out degree $k - 1$ exists takes time $O(k^2)$. In total, this takes time $O((T - \Delta)(k(\min(m, k^2)\log\Delta + k\log k) + k^2) = O(k(T - \Delta)(\min(m, k^2)\log\Delta + k\log k))$. ◀

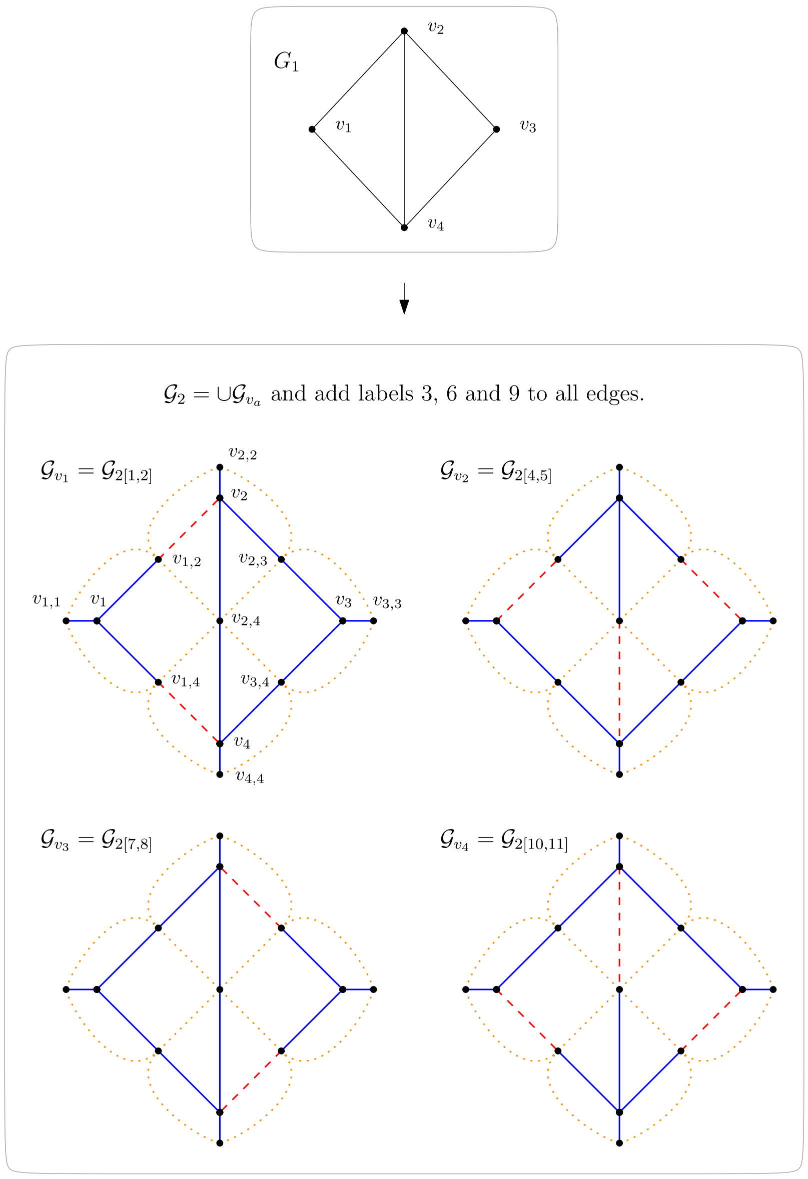
G_1
v_1
v_2
v_3
v_4
$\mathcal{G}_2 = \cup\mathcal{G}_{v_a}$ and add labels 3, 6 and 9 to all edges.
$\mathcal{G}_{v_1} = \mathcal{G}_{2[1,2]}$
$v_{2,2}$
$v_{1,2}$
$v_{2,3}$
$v_{1,1}$
$v_{2,4}$
$v_{3,3}$
$v_{1,4}$
$v_{3,4}$
$v_{4,4}$
$\mathcal{G}_{v_2} = \mathcal{G}_{2[4,5]}$
$\mathcal{G}_{v_3} = \mathcal{G}_{2[7,8]}$
$\mathcal{G}_{v_4} = \mathcal{G}_{2[10,11]}$

Fig. 7. Example of transformation from a CLIQUE instance graph G_1 to a $\mathcal{S}^{\mathcal{D}}$ COMPONENT instance temporal graph $\mathcal{G}_2$. For visibility, for all $\mathcal{G}_{v_a}$, blue full edges correspond to edges with label $3a-2$, red dashed edges to edges with label $3a-1$, and orange dotted edges to background edges with label $3a-1$. Also for visibility, not all background edges are represented.

Lemma 11. *A closed $\mathcal{S}^{\mathcal{D}}$ component of size $> \ell$ does not imply a (non-maximal) closed $\mathcal{S}^{\mathcal{D}}$ component of size ℓ exists.*

Proof. Consider Fig. 8, in which an infinite family of temporal graphs is presented ($T = \Delta + 1$), or each of these graphs, say $\mathcal{G}$, a closed $\mathcal{S}^{\mathcal{D}}$ component exists of size n, with any vertex as source for both windows. However, no (non-maximal) closed $\mathcal{S}^{\mathcal{D}}$ component exists of size $n-1$, since considering any such set of vertices V' would result in either $G(\mathcal{G}[V']_{[1,\Delta]})$ or $G(\mathcal{G}[V']_{[2,\Delta+1]})$ to be disconnected, implying no temporal source can exist in that window. ◀

This fact, common to other kinds of closed components, has a negative impact when searching for closed components. Indeed, if a candidate component over time interval $[1,\theta]$ is no more candidate at $\theta + 1$, its subsets have to be verified for all $t \le \theta$ (see also the proof of Theorem 9).

Theorem 14. *CLOSED $\mathcal{S}^{\mathcal{D}}$ COMPONENT is solvable in time $O(\min(2^n, n^{T-\Delta+1})(T-\Delta)n(m\log\Delta + n\log n))$.*

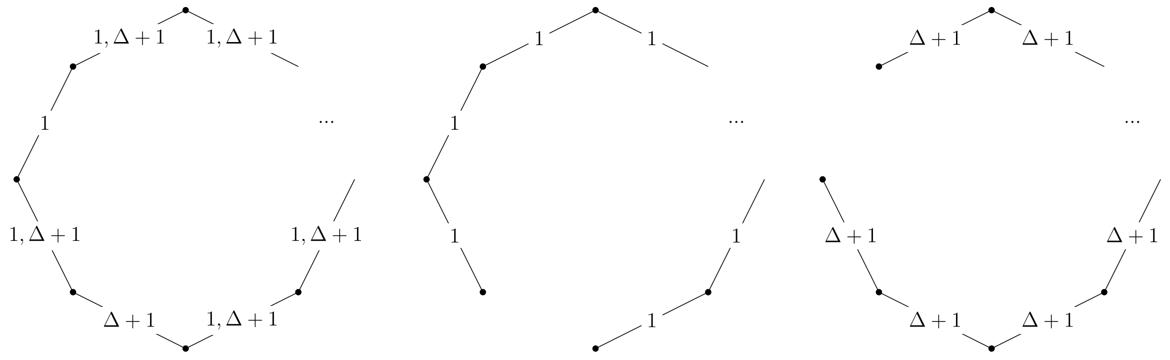

Fig. 8. Temporal graph family in which a closed $\mathcal{S}^{\mathcal{D}}$ component of size n exists but no non-maximal component of size $n-1$ exists. The temporal graphs corresponding to time window $[1, \Delta]$ and time window $[2, \Delta+1]$ are shown in the middle and on the right respectively.

Proof. Due to Lemma 11, a brute force algorithm cannot simply verify all possible subsets of size k, as a closed $\mathcal{S}^{\mathcal{D}}$ component of size $> k$ may still exist even if one of size k does not. Hence, all possible subsets of size at least k will have to be checked, of which there are $O(\min(2^n, n^{T-\Delta+1}))$ by Theorem 13. Verifying a solution takes time $O((T-\Delta)k(\min(m,k^2)\log\Delta + k\log k))$ by Lemma 10, meaning the brute force algorithm runs in time $O(\min(2^n, n^{T-\Delta+1})(T-\Delta)n(m\log\Delta + n\log n))$. ◀

Theorem 15. *Closed $\mathcal{S}^{\mathcal{D}}$ Component is NP-complete.*

Proof. To prove Closed $\mathcal{S}^{\mathcal{D}}$ Component is in NP, we need to show that if an instance is positive, then there exists a polynomial-size certificate which can be checked in polynomial time. We claim this certificate can be any (potentially non-maximal) closed $\mathcal{S}^{\mathcal{D}}$ component of size at least k, which can indeed be checked in polynomial time through Lemma 10.

To prove NP-hardness, the same reduction as in Theorem 12 is used, since all the $\mathcal{S}^{\mathcal{D}}$ components in the transformed instance of the reduction are closed $\mathcal{S}^{\mathcal{D}}$ components. ◀

4. All to all / all from all ($\mathcal{TC}$, $\mathcal{TC}^{\mathcal{B}}$, $\mathcal{TC}^{\circlearrowleft}$)

A graph is said to be temporally connected if there exists a journey from each vertex to every other vertex. In other words, over time, each vertex is able to connect to the whole temporal graph, and is used for example in contexts with multi-hop message passing, distributed mobile agents, or social communication networks [30,35,36]. Contrasting the previous section, finding all-to-all components is always NP-hard, which seems due to the potentially high number of connected components, and the closeness to finding cliques in the reachability graph. We consider fixed and constant lifetime T, and give tight results on when exactly the problems become hard regarding T.

Definition 7 *($\mathcal{TC}$ component).* A $\mathcal{TC}$ component of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\forall u, v \in V'$, $u \rightsquigarrow v$.

Similar to $\mathcal{S}^{\mathcal{B}}$, time windows add multiple advantages to temporal connectivity. Gomez et al. suppose a $\mathcal{TC}^{\mathcal{B}}$ component exists in their work on agreement in dynamic systems in [27]. In [14], Casteigts et al. present a general framework for computing parameters in temporal graphs, one of which being Δ for which the given graph is $\mathcal{TC}^{\mathcal{B}}$. Closed $\mathcal{TC}^{\mathcal{B}}$ components are studied by Huyghues-Despointes et al. in [32]. They propose polynomial-time algorithms for computing lower and upper bounds on the maximum component size.

Definition 8 *($\mathcal{TC}^{\mathcal{B}}$ component).* A $\mathcal{TC}^{\mathcal{B}}$ component of duration Δ of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\forall v, v' \in V'$, $\forall t \leq T - \Delta + 1$, $v \rightsquigarrow v'$ in temporal graphs $\mathcal{G}_{[t,t+\Delta-1]}$.

Another parameter Casteigts et al. are interested in is the round trip temporal diameter, being the shortest duration for which there exist round trip journeys between every pair of vertices. More generally, round-trip connectivity can represent systems in which feedback or acknowledgements are needed in a connection, such as Transmission Control Protocol (TCP).

Definition 9 *($\mathcal{TC}^{\circlearrowleft}$ component).* A $\mathcal{TC}^{\circlearrowleft}$ component of a temporal graph $\mathcal{G}$ is a maximal subset $V' \subseteq V$ such that $\forall u, v \in V', u \circlearrowleft v$ in $\mathcal{G}$.

4.1. $\mathcal{TC}$ components

Observation 4. *$\mathcal{TC}$ components can be open such as shown in Fig. 9.*

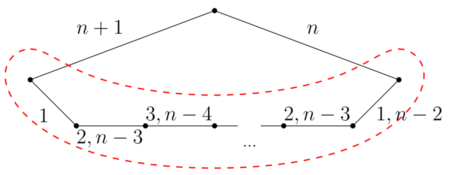

Fig. 9. A graph family admitting an open $\mathcal{TC}$ component (in dashed and red).

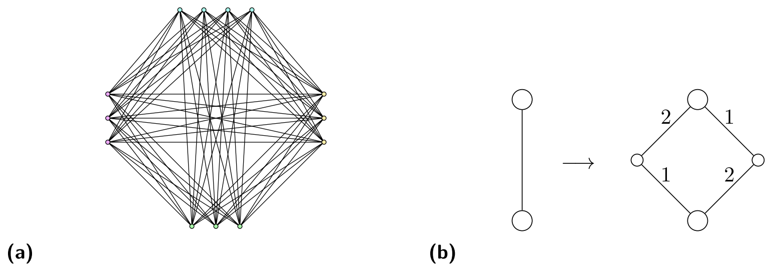

Fig. 10. A graph with at least $2^{0.52\sqrt{n}}$ $\mathcal{TC}$ components is constructed by taking a Moon and Moser graph (left) admitting $3^{n/3}$ cliques, and replacing each edge with a semaphore gadget (right). All initial cliques now correspond to $\mathcal{TC}$ components, which are maximal when including semaphore vertices.

As usual, we start with the general case: components may be open.

4.1.1. General case

A lower bound on the worst-case number of $\mathcal{TC}$ components has been obtained by Casteigts et al. [12] by adapting a Moon and Moser graph [41] on n vertices and $\binom{n}{2} - n$ edges admitting $3^{n/3}$ cliques. Every edge of this graph is replaced by a semaphore gadget (see Fig. 10) creating a temporal graph of $N = n^2 - 2n$ vertices in which every initial clique is now a $\mathcal{TC}$ component, obtaining a total of $3^{n/3} = 3^{(\sqrt{N+1}+1)/3} > 3^{\sqrt{N}/3} > 2^{0.52\sqrt{N}}$ $\mathcal{TC}$ components. We provide an upper bound on this number, to obtain the theorem below.

Theorem 16. *The worst-case number of $\mathcal{TC}$ components is at least $2^{0.52\sqrt{n}}$, and at most $2^{0.53n}$.*

Proof. Outside of the trivial upper bound of 2^n, a better upper bound can be obtained as follows. Since $\mathcal{TC}$ components of $\mathcal{G}$ correspond to bidirectional cliques in the reachability graph of $\mathcal{G}$ (see [9]), modify the reachability graph as follows. For all arcs (u, v) such that (v, u) exists as well, replace both arcs by one (undirected) edge $\{u, v\}$. All other arcs are removed. The corresponding graph is undirected and cliques in this graph correspond to $\mathcal{TC}$ components. According to Moon and Moser [41], at most $3^{n/3}$ cliques can exist in undirected graphs, which implies the same upper bound $3^{n/3} < 2^{0.53n}$ holds for $\mathcal{TC}$ components in temporal graphs. ◀

Note that the exact bounds are $3^{(\sqrt{n+1}+1)/3}$ and $3^{n/3}$ respectively. We used $2^{0.52k} < 3^{k/3} < 2^{0.53k}$ so that results are comparable with results using powers of 2.

An algorithm is mentioned by Bhadra et al. [9], and implemented and experimented on by Nicosia et al. [43]. To the best of our knowledge, no complexity analysis of this algorithm has been performed.

Theorem 17. *$\mathcal{TC}$ COMPONENT is solvable in time $O(nm\log T + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$.*

Proof. The algorithm starts by constructing the reachability graph of the input temporal graph. This is done in time $O(n(m\log T + n\log n))$. Then, it searches for bidirectional cliques (a subset of vertices with arcs in both directions between vertices) in the reachability graph. This has the same asymptotic complexity as searching for cliques. Since we're interested in cliques of size at least k, the brute force algorithm of testing each subset of size k runs in time $O(\frac{n^k k^2}{k!})$. Also, one can obtain a maximum-size clique in time $O(2^{n/4})$ [45]. All together, $\mathcal{TC}$ COMPONENT can thus be solved in time $O(n(m\log T + n\log n) + \min(\frac{n^k k^2}{k!}, 2^{n/4})) = O(nm\log T + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$. ◀

The good news is that a polynomial algorithm exists when k is fixed. Unfortunately, Bhadra et al. prove that $\mathcal{TC}$ COMPONENT is NP-complete. They use a reduction from CLIQUE in which they produce a temporal graph with a lifetime $T = 4$ (so using labels between 1 and 4 included). This leaves the NP-hardness question open for lifetimes $1 < T < 4$ ($T = 1$ being polynomial-time solvable). We treat the case of $T = 2$, thus covering the gap. We believe Bhadra et al. may have attempted to reduce using only the semaphore trick (thus with $T = 2$) but encountered some issues concerning large $\mathcal{TC}$ components in the transformed instance even when no large cliques are present in the CLIQUE instance. They proposed a solution to this issue using their structure requiring two extra time steps. We first discovered a solution to this by using "elongated semaphores", *i.e.* semaphores with two intermediary vertices on each

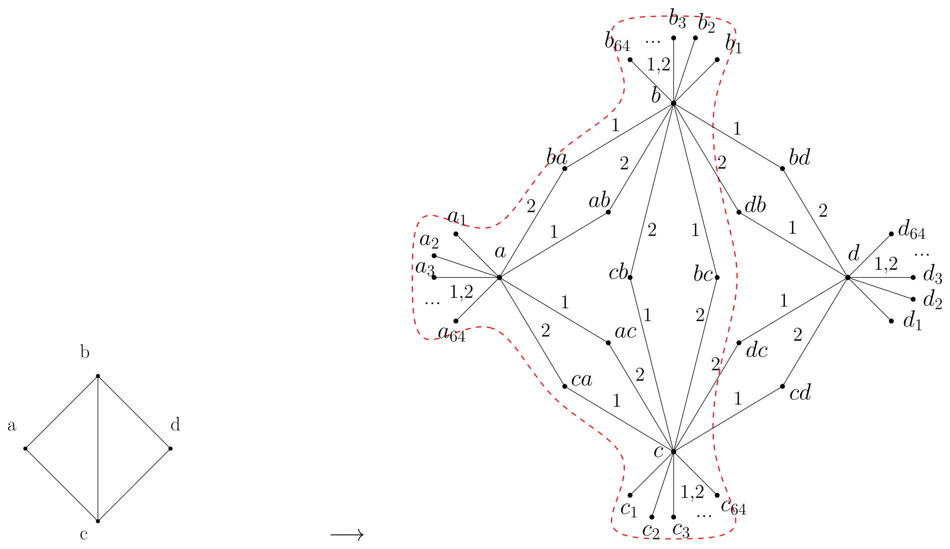
b
a
d
c
b_{64} $\cdots$ b_3 b_2 b_1
1,2
b
ba
ab
db
bd
a_1 a_2 a_3 $\cdots$ a_{64}
a
cb
bc
d
d_{64} $\cdots$ d_3 d_2 d_1
ac
dc
ca
cd
c
c_1 c_2 c_3 $\cdots$ c_{64}
1
2
$\longrightarrow$

Fig. 11. On the left an example instance of CLIQUE with a clique of size 3 which, when transformed as shown on the right, allows for a $\mathcal{TC}$ component of size 201 (shown in red and dashed).

journey instead of only one, and using labels 1, 2 and 3 instead of 1 and 2. We present here a better solution, being the satellite vertices, which allows for the temporal graph to remain at lifetime $T = 2$. This result complements [18], where the authors prove $W[1]$-hardness of the problem with unbounded lifetime, and of the problem in directed temporal graphs with a bounded lifetime of $T = 2$.

Theorem 18. *$\mathcal{TC}$ COMPONENT is NP-complete, for all constant lifetimes $T > 1$.*

Proof. First, $\mathcal{TC}$ COMPONENT is in NP because a given set of vertices $V' \subseteq V$ can be checked in polynomial time to be a (potentially non-maximal) $\mathcal{TC}$ component: it is sufficient to construct the reachability graph and to verify if a bidirectional clique exists between vertices V'.

For NP-hardness, we show how to reduce CLIQUE to $\mathcal{TC}$ COMPONENT in polynomial time. Given a graph $G = (V, E)$ for the CLIQUE problem, create temporal graph $\mathcal{G}$ with (initially) the same vertex set V. For each edge $(u, v) \in E$, add a semaphore gadget between u and v in $\mathcal{G}$, with intermediary vertices uv and vu. Also, add n^3 satellite vertices $v_{i,1}, \ldots, v_{i,n^3}$ to every original vertex v_i, with the corresponding edges having labels 1 and 2. See Fig. 11 for an example of how to construct $\mathcal{G}$. Note that $\mathcal{G}$ has lifetime $T = 2$.

We prove a clique of size k exists in G if and only if a $\mathcal{TC}$ component of size $kn^3 + k^2$ exists in $\mathcal{G}$.

CLIQUE $\implies$ $\mathcal{TC}$ COMPONENT
If a clique of size at least k exists in G of vertices $K = \{v_i : i \leq |K|\}$, then consider the corresponding vertices in $\mathcal{G}$, as well as all corresponding intermediary vertices between them (these must exist by construction), and all corresponding satellite vertices. Let's refer to this set of vertices in $\mathcal{G}$ as V'. V' is of size $kn^3 + k^2$ and furthermore V' is a $\mathcal{TC}$ component, since all the vertices can reach each other: any vertex $v_i \in K$ can clearly reach any other vertex $v_j \in K$ through the semaphore between the vertices. Concerning the other vertices in V', since they all have an edge with label 1 (*resp.* label 2) connecting them to a vertex $v_i \in K$, they can reach (*resp.* be reached by) the exact same vertex set v_i can reach (*resp.* be reached by). Thus, a $\mathcal{TC}$ component of size at least $kn^3 + k^2$ must exist.

$\mathcal{TC}$ COMPONENT $\implies$ CLIQUE
If a (maximal) $\mathcal{TC}$ component V' of size at least $kn^3 + k^2$ exists in $\mathcal{G}$, then consider all vertices $K = V' \cap V$. Clearly $|K| \geq k$ as there exists no other way of obtaining an addend of at least kn^3 in the size of V' than it containing at least k groups of satellite vertices, and by maximality if a satellite vertex $v_{i,a}$ is contained in V', then so must all other satellite vertices $v_{i,b}$ for all b, as well as the original vertex v_i. Since all vertices in K are part of a $\mathcal{TC}$ component V', they are all able to reach each other, which is only possible if a semaphore gadget exists between each pair of vertices of K. A semaphore gadget can only be present if a corresponding edge exists in G, thus K forms a clique of size at least k in G, finishing the proof. ◀

4.1.2. Closed $\mathcal{TC}$ components

Theorem 19. *The worst-case number of closed $\mathcal{TC}$ components is at least $2^{0.52\sqrt{n}}$, and at most $2^{0.53n}$.*

Proof. Theorem 16 directly adapts for closed components, as the components obtained by the lower bound's construction are closed. ◀

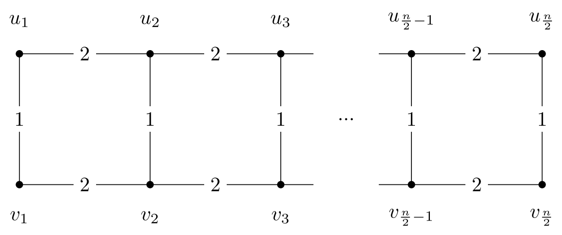

Fig. 12. Temporal graph family in which a closed $\mathcal{TC}$ component of size n exists but no non-maximal component of size $n-1$ exists.

Lemma 12. *Verifying if a vertex subset V' of size k is a (potentially non-maximal) closed $\mathcal{TC}$ component takes time $O(k(\min(m,k^2)\log T + k\log k))$.*

Proof. To verify V', we construct the reachability graph of temporal graph $\mathcal{G}[V']$ and check if it is a (bidirectional) complete graph, doable in time $O(k(\min(m,k^2)\log T + k\log k) + k^2) = O(k(\min(m,k^2)\log T + k\log k))$. ◀

Observation 5. *A closed $\mathcal{TC}$ component of size $> k$ does not imply a (non-maximal) closed $\mathcal{TC}$ component of size k exists.*

Proof. Consider Fig. 12, in which an infinite family of temporal graphs is presented. For each of these graphs, say $\mathcal{G}$, a closed $\mathcal{TC}$ component exists of size n. However, no (non-maximal) closed $\mathcal{TC}$ component exists of size $n-1$, since considering any such set of vertices $V \setminus u_i$ (*resp.* v_i) would result in v_i (*resp.* u_i) not being able to reach any vertex v_j (*resp.* u_j). ◀

Theorem 20. *Closed $\mathcal{TC}$ Component is solvable in time $O(2^n n(m\log T + n\log n))$.*

Proof. We describe the brute force algorithm of testing all possible subsets of vertices. Due to Observation 5, we cannot only test subsets of size k, needing to possibly test all subsets of size at least k, of which there are $O(2^n)$. Verifying a vertex subset takes time $O(n(m\log T + n\log n))$ by Lemma 12, giving a total time complexity of $O(2^n n(m\log T + n\log n))$. ◀

Bhadra et al. also prove Closed $\mathcal{TC}$ Component is NP-complete through the same reduction as for $\mathcal{TC}$ Component, again leaving open NP-hardness for constant lifetimes $1 < T < 4$. We completely fill the gap, by proving NP-hardness for $T = 2$.

Theorem 21. *Closed $\mathcal{TC}$ Component is NP-complete, for all constant lifetimes $T > 1$.*

Proof. To prove Closed $\mathcal{TC}$ Component is in NP, we need to show that if an instance is positive, then there exists a polynomial-size certificate which can be checked in polynomial time. We claim this certificate can be any (potentially non-maximal) closed $\mathcal{TC}$ component of size at least k, which can indeed be checked in polynomial time through Lemma 12.

In the construction from Theorem 18, all $\mathcal{TC}$ components (of required size) in $\mathcal{G}$ are closed. This suffices to prove that Closed $\mathcal{TC}$ Component is NP-hard as well, for all lifetimes $T > 1$. Since Closed $\mathcal{TC}$ Component is both in NP and NP-hard, it is thus NP-complete. ◀

4.2. $\mathcal{TC}^{\mathcal{B}}$ components

Lemma 13. *$\mathcal{TC}^{\mathcal{B}}$ components can be open.*

Proof. Since $\mathcal{TC}$ components are $\mathcal{TC}^{\mathcal{B}}$ components for the specific setting of $\Delta = T$, Theorem 16 transfers directly. ◀

4.2.1. General case

Theorem 22. *The worst-case number of $\mathcal{TC}^{\mathcal{B}}$ components is at least $2^{0.52\sqrt{n}}$ and at most $2^{0.53n}$.*

Proof. The lower bound transfers directly from Theorem 16.

The upper bound is obtained in a similar manner as for Theorem 16. Start by creating an adaptation of the reachability graph, in which an arc (u,v) is present if and only if a journey exists from u to v in every window of size Δ. (This can be computed easily using temporal Dijkstra algorithm from [48].) Then, for all arcs (u,v) such that (v,u) exists as well, replace both arcs by one (undirected) edge $\{u,v\}$. All other arcs are removed. The corresponding graph is non directed and cliques in this graph correspond to $\mathcal{TC}^{\mathcal{B}}$ components. According to Moon and Moser [41], at most $3^{n/3}$ cliques can exist in non directed graphs, which implies the same upper bound $3^{n/3} < 2^{0.53n}$ holds for $\mathcal{TC}^{\mathcal{B}}$ components in temporal graphs. ◀

Theorem 23. *$\mathcal{TC}^{\mathcal{B}}$ Component is solvable in time*

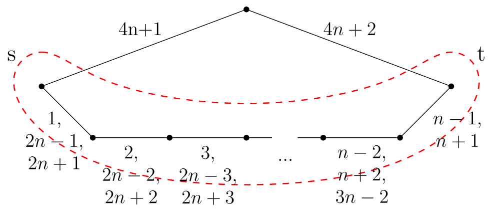

Fig. 13. Graph family admitting an open $\mathcal{TC}^{\circlearrowleft}$ component (in red and dashed). Vertex s can reach vertex t and back through a journey inside the component, but a journey from t to s and back has to go outside the component.

Proof. For all windowed temporal graphs $\mathcal{G}_{[t,t+\Delta-1]}$, construct reachability graphs $H_{[t,t+\Delta-1]}$, taking time $O((T-\Delta)(n(m\log\Delta + n\log n)))$. Afterwards, take the intersection of these reachability graphs, denoted by H. An arc (u,v) in H means that u can reach v in all windowed temporal graphs. A bidirectional clique of size k in H thus represents a $\mathcal{TC}^{\mathcal{B}}$ component of size k. As described in Theorem 17, such a clique of size at least k can be found in time $O(\min(\frac{n^k k^2}{k!}, 2^{n/4}))$. All in all, $\mathcal{TC}^{\mathcal{B}}$ Component can thus be solved in time $O((T-\Delta)(n(m\log\Delta + n\log n)) + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$. ◀

Again, since $\mathcal{TC}$ components are $\mathcal{TC}^{\mathcal{B}}$ components by setting window size $\Delta = T$, NP-hardness transfers. We however prove a more general hardness for $\mathcal{TC}^{\mathcal{B}}$ Component regarding any constant lifetime and window size.

Theorem 24. *$\mathcal{TC}^{\mathcal{B}}$ Component is NP-complete, for all constant lifetimes and window sizes $T \geq \Delta > 1$.*

Proof. It is trivially true that instances of $\mathcal{TC}$ Component are equivalent to instances of $\mathcal{TC}^{\mathcal{B}}$ Component such that $\Delta = T$.

Let us consider Δ arbitrarily smaller than T. Any instance $(\mathcal{G}, k)$ for $\mathcal{TC}$ Component, where the lifetime of $\mathcal{G}$ is T, is equivalent to instance $(\mathcal{G}', k, \Delta = T)$ for $\mathcal{TC}^{\mathcal{B}}$ Component, where $\mathcal{G}'$ corresponds to $\mathcal{G}$ with some additional complete snapshots, *i.e.* $\mathcal{G}' = \mathcal{G} \cup (G_i = K_n)_{i=T+1}^{T'}$ for some arbitrary $T' > T$. Combined with Theorem 18, this proves that $\mathcal{TC}^{\mathcal{B}}$ Component is NP-hard for all $T \geq \Delta > 1$. ◀

4.2.2. Closed $\mathcal{TC}^{\mathcal{B}}$ components

Theorem 25. *The worst-case number of closed $\mathcal{TC}^{\mathcal{B}}$ components is at least $2^{0.52\sqrt{n}}$ and at most $2^{0.53n}$.*

Proof. The lower bound from Theorem 16 transfers since the construction produces closed $\mathcal{TC}$ components, which are also closed $\mathcal{TC}^{\mathcal{B}}$ components for $\Delta = T$. The upper bounds from Theorem 22 transfer directly. ◀

Lemma 14. *Verifying if a vertex subset V' of size k is a (potentially non-maximal) closed $\mathcal{TC}^{\mathcal{B}}$ component takes time $O((T-\Delta)k(\min(m,k^2)\log\Delta + k\log k))$.*

Proof. For each window $[t, t+\Delta-1]$, compute the reachability graph $H_{[t,t+\Delta-1]}$ of temporal graph $\mathcal{G}[V']_{[t,t+\Delta-1]}$. V' is a solution for Closed $\mathcal{TC}^{\mathcal{B}}$ Component if and only if all $H_{[t,t+\Delta-1]}$ are bidirectional complete graphs. ◀

Theorem 26. *Closed $\mathcal{TC}^{\mathcal{B}}$ Component is solvable in time $O(2^n(T-\Delta)n(m\log\Delta + n\log n))$.*

Proof. We present the brute force algorithm. Due to Observation 5, we know it may not be sufficient to verify only the subsets of size exactly k; we may need to verify all subsets of size at least k, of which there exist $O(2^n)$. By Lemma 14, verifying such a subset requires $O((T-\Delta)n(m\log\Delta + n\log n))$ time. In total, the brute force algorithm thus takes time $O(2^n(T-\Delta)n(m\log\Delta + n\log n))$. ◀

Theorem 27. *Closed $\mathcal{TC}^{\mathcal{B}}$ Component is NP-complete, for all constant lifetimes and window sizes $T \geq \Delta > 1$.*

Proof. To prove Closed $\mathcal{TC}^{\mathcal{B}}$ Component is in NP, we need to show that if an instance is positive, then there exists a polynomial-size certificate which can be checked in polynomial time. We claim this certificate can be any (potentially non-maximal) closed $\mathcal{TC}^{\mathcal{B}}$ component of size at least k, which can indeed be checked in polynomial time through Lemma 14.

In the reductions used in Theorems 18 and 24, all $\mathcal{TC}^{\mathcal{B}}$ components in the transformed instance are closed. This suffices to prove that Closed $\mathcal{TC}^{\mathcal{B}}$ Component is NP-hard, for all constant lifetimes and window sizes $T \geq \Delta > 1$. ◀

4.3. $\mathcal{TC}^{\circlearrowleft}$ components

Observation 6. *$\mathcal{TC}^{\circlearrowleft}$ components can be open, such as shown in Fig. 13.*

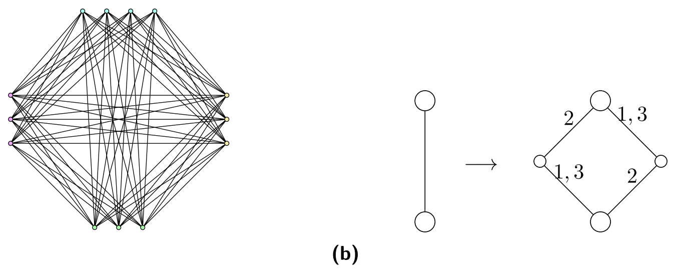

Fig. 14. A graph with at least $2^{0.52\sqrt{n}}$ $\mathcal{TC}^{\circlearrowleft}$ components is constructed by taking a Moon and Moser graph (left) admitting $3^{n/3}$ cliques, and replacing each edge with a semaphore gadget (right). All initial cliques now correspond to $\mathcal{TC}^{\circlearrowleft}$ components, which are maximal when including semaphore vertices.

Definition 10 *(Extension of reachability graph).* $H'(\mathcal{G}) = (V, E')$ is the directed graph obtained as follows: $\forall u, v \in V$, $(u, v) \in E' \Leftrightarrow u \circlearrowleft v$.
$H'(\mathcal{G}) = (V, E')$ is denoted H' when there is no ambiguity.

Lemma 15. *Constructing graph $H'(\mathcal{G})$ takes time $O(n^2(m \log T + n \log n))$.*

Proof. Initialize H' with vertex set V. For all couples of vertices, of which there are $O(n^2)$, do the following. Apply the temporal Dijkstra algorithm from [48] starting from u to obtain the smallest label at which v is reached from u through a journey, suppose label t. If v is unreachable from u, let $t = \infty$. This takes time $O(m \log T + n \log n)$. Then, if $t \neq \infty$, on temporal graph $\mathcal{G}_{[t,T]}$, apply again the temporal Dijkstra algorithm to check if u is reachable from v. This again takes time $O(m \log T + n \log n)$. If $t \neq \infty$ and if u is reachable from v in $\mathcal{G}_{[t,T]}$, then add arc (u, v) to H. ◀

4.3.1. General case

Theorem 28. *The worst-case number of $\mathcal{TC}^{\circlearrowleft}$ components is at least $2^{0.52\sqrt{n}}$, and at most $2^{0.53n}$.*

Proof. The lower bound can be obtained as follows. Take a Moon and Moser graph [41] on n vertices and $\binom{n}{2} - n$ edges admitting $3^{n/3}$ cliques. Every edge of this graph is replaced by a semaphore gadget (see Fig. 14) creating a temporal graph of $N = n^2 - 2n$ vertices in which every initial clique is now a $\mathcal{TC}^{\circlearrowleft}$ component, obtaining a total of $3^{n/3} > 2^{0.52\sqrt{N}}$ $\mathcal{TC}^{\circlearrowleft}$ components.

Outside of the trivial upper bound of 2^n, a better upper bound can be obtained as follows. By definition, $\mathcal{TC}^{\circlearrowleft}$ components of $\mathcal{G}$ are exactly the bidirectional cliques of H'. Further modify H' as follows. For all arcs (u, v) such that (v, u) exists as well, replace both arcs by one (non directed) edge $\{u, v\}$. All other arcs are removed. The corresponding graph is non directed and cliques in this graph correspond to $\mathcal{TC}^{\circlearrowleft}$ components. According to Moon and Moser [41], at most $3^{n/3}$ cliques can exist in non directed graphs, which implies the same upper bound $3^{n/3} < 2^{0.53n}$ holds for $\mathcal{TC}^{\circlearrowleft}$ components in temporal graphs. ◀

Lemma 16. *Verifying if a vertex subset V' of size k is a solution for $\mathcal{TC}^{\circlearrowleft}$ COMPONENT takes time $O(k^2(m \log T + n \log n))$.*

Proof. As established in Lemma 15 proof, checking if $u \circlearrowleft v$ in $\mathcal{G}$ is $O(m \log T + n \log n)$. As there are $O(k^2)$ couples of vertices to verify, the result follows. ◀

Note that another way to check this is to build H', and then simply check if V' is a bidirectional clique, which is $O(k^2)$. However, see Lemma 15, building H' has larger complexity than the algorithm above.

Theorem 29. *$\mathcal{TC}^{\circlearrowleft}$ COMPONENT is solvable in time $O(n^2(m \log T + n \log n) + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$.*

Proof. Let us first construct H'. Then modify it such that if (u, v) and (v, u) exist, replace both arcs by a (non directed) edge $\{u, v\}$. Remove all other arcs. The resulting graph is now non directed, and its cliques correspond to $\mathcal{TC}^{\circlearrowleft}$ components. Since we are interested in cliques of size at least k, the brute force algorithm of testing each subset of size k runs in time $O(\frac{n^k k^2}{k!})$ (the number of subsets is $O(\frac{n^k}{k!})$). Also, one can obtain a maximum-size clique in time $O(2^{n/4})$ [45]. All together, $\mathcal{TC}^{\circlearrowleft}$ COMPONENT can thus be solved in time $O(n^2(m \log T + n \log n) + \min(\frac{n^k k^2}{k!}, 2^{0.25n}))$. ◀

Note that using the algorithm of Lemma 16 to check directly each subset is exactly $O(\frac{n^k k^2}{k!}(m \log T + n \log n))$. The reader can verify that this complexity is strictly higher than that of the theorem as soon as $k > 3$.

Outside of the trivial case of lifetime $T = 1$, $\mathcal{TC}^{\circlearrowleft}$ COMPONENT can be solved in polynomial time for lifetime $T = 2$ as well.

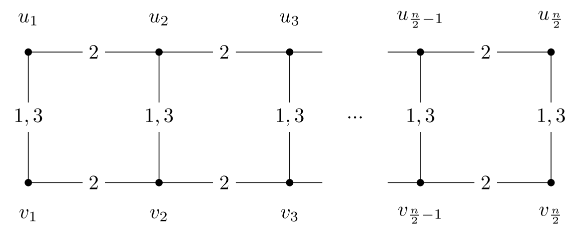

Fig. 15. Temporal graph family in which a closed $\mathcal{TC}^{\circlearrowleft}$ component of size n exists but no non-maximal component of size $n-1$ exists.

Theorem 30. *$\mathcal{TC}^{\circlearrowleft}$ COMPONENT is solvable in polynomial time $O(n+m)$ on temporal graphs with constant lifetimes $T \leq 2$.*

Proof. The case of $T = 1$ trivially reduces to finding connected components in the snapshot, doable in time $O(n+m)$.

For the case of $T = 2$, note that three possible $\mathcal{TC}^{\circlearrowleft}$ components can exist. The first (*resp.* second) type of $\mathcal{TC}^{\circlearrowleft}$ components uses only labels 1 (*resp.* 2). These $\mathcal{TC}^{\circlearrowleft}$ components correspond to connected components in the corresponding snapshot, and can thus be computed in time $O(n+m)$ as well. The third, and final, type of $\mathcal{TC}^{\circlearrowleft}$ components uses both labels 1 and 2. We continue by proving the latter type of $\mathcal{TC}^{\circlearrowleft}$ components does not exist, finishing the proof.

By contradiction, let us suppose one such a $\mathcal{TC}^{\circlearrowleft}$ component exists, say V'. For any two vertices $u, v \in V'$, if a journey using only labels 1 (*resp.* 2) does not exist between u and v, then a journey using only labels 2 (*resp.* 1) must exist between v and u (or V' would not be a $\mathcal{TC}^{\circlearrowleft}$ component). Let us build the non directed graph $G' = (V', E')$, such that an edge with label i is drawn between vertices u and v if the corresponding journey between u and v uses only labels i. G' is thus a complete graph with some arbitrary labelling of one label per edge, being either 1 or 2. We finish by showing that in G' there is necessarily a spanning structure using only label 1, or using only label 2, implying that in fact V' is not a $\mathcal{TC}^{\circlearrowleft}$ component needing both labels 1 and 2, as a spanning structure using only labels 1 or only labels 2 exists which is sufficient for $\mathcal{TC}^{\circlearrowleft}$.

G contains a spanning structure using only labels 1 or only labels 2:
We prove this property by induction. A K_2 graph trivially admits such a spanning structure. Now, if a K_n graph has this property, suppose *w.l.o.g.* that the spanning structure has only labels 1 (the case of the spanning structure having only labels 2 is analogous by switching labels 1 and 2). Adding a vertex so as to obtain a K_{n+1} implies adding n edges connecting this vertex to the other vertices. To avoid having this property in the newly constructed K_{n+1}, none of these edges can have label 1 (since otherwise a spanning structure with only labels 1 exists), but they cannot all have label 2 either (since otherwise a spanning structure with only labels 2 exists), which is impossible. ◀

Theorem 31. *$\mathcal{TC}^{\circlearrowleft}$ COMPONENT is NP-complete, for all constant lifetimes $T > 2$.*

Proof. We show how to reduce CLIQUE to $\mathcal{TC}^{\circlearrowleft}$ COMPONENT in polynomial time. Given a graph $G = (V, E)$ for the CLIQUE problem, create temporal graph $\mathcal{G}$ such as specified in Theorem 18. See again Fig. 11 for an example of how to construct $\mathcal{G}$. Then, for each edge having label 1, also add label 3. Note that $\mathcal{G}$ has lifetime $T = 3$.

We prove a clique of size k exists in G if and only if a $\mathcal{TC}^{\circlearrowleft}$ component of size $kn^3 + k^2$ exists in $\mathcal{G}$.

Clique $\implies$ $\mathcal{TC}^{\circlearrowleft}$ COMPONENT
If a clique of size at least k exists in G of vertices $K = \{v_i : i \leq |K|\}$, then consider the corresponding vertices in $\mathcal{G}$, as well as all corresponding intermediary vertices between them (which must exist by construction), and all corresponding satellite vertices. Let's refer to this set of vertices in $\mathcal{G}$ as V', which is of size $kn^3 + k^2$. V' is a $\mathcal{TC}^{\circlearrowleft}$ component, since all vertices can reach each other and then reach back: any vertex $v_i \in K$ can clearly reach any other vertex $v_j \in K$ and back, through the semaphore between the vertices and back. Note that the reaching of v_j (before heading back) necessarily happens at time 2. Concerning every other vertex v_k in V', since they all have an edge with labels 1 and 3 connecting to a vertex $v_i \in K$, we have that for all vertices v_ℓ such that $v_i \circlearrowleft v_\ell$, $v_k \circlearrowleft v_\ell$. Furthermore, since v_k has an edge with label 2 connecting to some $v_i \in K$, we have that for all vertices v_ℓ such that $v_\ell \circlearrowleft v_i$, $v_\ell \circlearrowleft v_k$. Thus, a $\mathcal{TC}^{\circlearrowleft}$ component of size at least $kn^3 + k^2$ must exist.

$\mathcal{TC}^{\circlearrowleft}$ COMPONENT $\implies$ CLIQUE
If a (maximal) $\mathcal{TC}^{\circlearrowleft}$ component V' of size at least $kn^3 + k^2$ exists in G, then consider all vertices $K = V' \cap V$. Clearly $|K| \geq k$ as there exists no other way of obtaining an addend of at least kn^3 in the size of V' than it containing at least k groups of satellite vertices, and by maximality if a satellite vertex $v_{i,a}$ is contained in V', then so must all other satellite vertices $v_{i,b}$ for all b, as well as the original vertex v_i, since they are all connected by edges with all labels (effectively acting like one massive vertex). Since all vertices in K are part of a $\mathcal{TC}^{\circlearrowleft}$ component V', they are all able to reach each other and back, which is only possible if a semaphore gadget (with the added label 3) exists between each pair of vertices of K. Such a semaphore gadget can only be present if a corresponding edge exists in G, thus K forms a clique of size at least k in G, finishing the proof. ◀

4.3.2. Closed $\mathcal{TC}^{\circlearrowleft}$ components

Theorem 32. *The worst-case number of closed $\mathcal{TC}^{\circlearrowleft}$ components is at least $2^{0.52\sqrt{n}}$, and at most $2^{0.53n}$.*

Proof. The lower bound comes from the fact that all $\mathcal{TC}^{\circlearrowleft}$ components in the construction from Theorem 28 are closed, and the upper bound transfers directly from Theorem 28. ◀

Lemma 17. *Verifying if a vertex subset V' of size k is a (potentially non-maximal) closed $\mathcal{TC}^{\circlearrowleft}$ component takes time $O(k^2(\min(m,k^2)\log T + k\log k))$.*

Proof. Let us consider the induced subgraph $\mathcal{G}[V']$, with k vertices and m' edges, and the corresponding adapted reachability graph $H'(\mathcal{G}[V']) = H'$. According to Lemma 15, H' can be computed in $O(k^2(m'\log T + k\log k))$. Checking that H' is a bidirectional clique is $O(k^2)$. As $m' = O(\min(m,k^2)$, the result follows. ◀

Lemma 18. *A closed $\mathcal{TC}^{\circlearrowleft}$ component of size $> k$ does not imply a (non-maximal) closed $\mathcal{TC}^{\circlearrowleft}$ component of size k exists.*

Proof. Consider Fig. 15, in which an infinite family of temporal graphs is presented. For each of these graphs, say $\mathcal{G}$, a closed $\mathcal{TC}^{\circlearrowleft}$ component exists of size n. However, no (non-maximal) closed $\mathcal{TC}^{\circlearrowleft}$ component exists of size $n-1$, since considering any such set of vertices $V \setminus u_i$ (*resp.* v_i) would result in v_i (*resp.* u_i) not being able to reach any vertex v_j (*resp.* u_j) and back. ◀

Theorem 33. *Closed $\mathcal{TC}^{\circlearrowleft}$ Component is solvable in time $O(2^n(n^2(m\log T + n\log n))$.*

Proof. We present the brute force algorithm. Due to Lemma 18, we know it may not be sufficient to verify only the subsets of size exactly k; we may need to verify all subsets of size at least k, of which there exist $O(2^n)$. By Lemma 17, verifying such a subset requires $O(n^2(m\log T + n\log n))$ time. In total, the brute force algorithm thus takes time $O(2^n(n^2(m\log T + n\log n))$. ◀

Theorem 34. *Closed $\mathcal{TC}^{\circlearrowleft}$ Component is solvable in polynomial time $O(n+m)$ on temporal graphs with constant lifetimes $T \leq 2$.*

Proof. In Theorem 30, we prove that all $\mathcal{TC}^{\circlearrowleft}$ components in temporal graphs of lifetime $T \leq 2$ correspond to connected components in G_1 or G_2. Such connected components are necessarily closed, thus all $\mathcal{TC}^{\circlearrowleft}$ components in temporal graphs of lifetime $T \leq 2$ are necessarily closed. Since $\mathcal{TC}^{\circlearrowleft}$ Component can be solved in time $O(m+n)$ on such temporal graphs, Closed $\mathcal{TC}^{\circlearrowleft}$ Component can be solved in time $O(m+n)$ as well. ◀

Theorem 35. *Closed $\mathcal{TC}^{\circlearrowleft}$ Component is NP-complete, for all constant lifetimes $T > 2$.*

Proof. To prove Closed $\mathcal{TC}^{\circlearrowleft}$ Component is in NP, we need to show that if an instance is positive, then there exists a polynomial-size certificate which can be checked in polynomial time. We claim this certificate can be any (potentially non-maximal) closed $\mathcal{TC}^{\circlearrowleft}$ component of size at least k, which can indeed be checked in polynomial time through Lemma 17.

In the reduction used in Theorem 31, all $\mathcal{TC}^{\circlearrowleft}$ components in the transformed instance are closed. This suffices to prove that Closed $\mathcal{TC}^{\circlearrowleft}$ Component is NP-hard, for any constant lifetime $T > 2$. ◀

5. Conclusion

5.1. General perspectives

This paper studies many extensions of connected components to temporal graphs, especially those defined thanks to journeys (temporal paths), distinguishing, when relevant, closed and general components. It provides results on the number of such components, the complexity of the problem to find them, and associated algorithms. It closes some complexity gaps by proposing new proofs. There is still room for improvement, particularly with regard to the maximum number of components and the algorithmic complexity. Furthermore, the paper did not consider strict-journey-based connectivity, nor path-based connectivity. Furthermore, no hypothesis is done on the dynamicity of the considered temporal graphs. If this dynamicity is low, better adapted data structures, such as keeping track of the changes between successive snapshots, or representing the temporal graph by a set of presence intervals, might help to design faster algorithms.

5.2. A short discussion about optimisation of windowed components

This paper provided algorithms (polynomial or not) to solve the windowed versions of our problems. Some of these consider the windows in the specific order $([1,\Delta],[2,\Delta+1],...[T-\Delta+1,T])$. We note this order isn't necessary, in the sense that the result would remain unchanged if the time windows were considered in any order. A consequence potentially useful for optimisation is that one could reorder the time windows in the most "favourable" way. For example, for $\mathcal{S}^{\mathcal{B}}$, start with the window in which the corresponding temporal graph may have few and small candidate components, and start the intersection process with windows in which the corresponding temporal graphs may have very distinct $\mathcal{S}$ components from the candidate components. Of course, theoretically this would only be useful if such an estimation and reordering of time windows could be done in time $O((T-\Delta)(n(m\log\Delta+n\log n)))$ as well, and even then practically this may still induce a significant change in running time.

Another possible optimisation concerns some bounded versions of problems related to $\mathcal{TC}$, where reachability graphs are computed for each time window. Can one do better by computing a modified reachability graph over the whole graph, which iteratively keeps track of how old journeys' starting dates are, and removes them over time if too old? Our first investigations did not allow to show this idea brings a breakthrough.

Both of these directions we leave as open research avenues in this paper.

5.3. A quick note on parameterised complexity

Among the NP-hard problems in this paper, all but two are para-NP-hard concerning the lifetime parameter, *i.e.* for some constant value of T the problem is NP-hard. The two problems which are not para-NP-hard regarding lifetime T are $\mathcal{S}^{\mathcal{D}}$ Component and Closed $\mathcal{S}^{\mathcal{D}}$ Component. Both are in XP, *i.e.* the problems are polynomial-time solvable for all constant T.

In the same manner, all general versions of the NP-hard problems are in XP regarding the size parameter k. So $\mathcal{S}^{\mathcal{D}}$ Component is in XP regarding both parameters. By contrast, all closed versions are likely to remain para-NP-hard concerning the size parameter. It may be of interest to determine where exactly in XP the problems are situated, *e.g.* FPT or W[1]-hard. Of course, outside of T and k, other natural parameters may be considered. Again, this research is left as open avenue.

Funding

Jason Schoeters: DyNet RIN Tremplin Région Normandie 2020-2022.

CRediT authorship contribution statement

Stefan Balev: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Investigation, Formal analysis, Conceptualization. **Eric Sanlaville:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Jason Schoeters:** Writing – review & editing, Writing – original draft, Visualization, Validation, Project administration, Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

We thank the editor and referees for their careful reading and constructive comments which significantly improved the presentation of these results.

References

[1] Eleni C. Akrida, Leszek Gasieniec, George B. Mertzios, Paul G. Spirakis, The complexity of optimal design of temporally connected graphs, Theory Comput. Syst. 61 (3) (2017) 907–944.
[2] Eleni C. Akrida, Paul G. Spirakis, On verifying and maintaining connectivity of interval temporal networks, Parallel Process. Lett. 29 (02) (2019), https://doi.org/10.1142/S0129626419500099.
[3] Aris Anagnostopoulos, Ravi Kumar, Mohammad Mahdian, Eli Upfal, Fabio Vandin, Algorithms on evolving graphs, in: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS '12, 2012, pp. 149–160.
[4] Brenda Baker, Robert Shostak, Gossips and telephones, Discrete Math. 2 (3) (June 1972) 191–193.
[5] Stefan Balev, Yoann Pigné, Eric Sanlaville, Mathilde Vernet, Complexité du problème de Steiner dynamique, in: 23ème Congrès Annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, Villeurbanne - Lyon, France, February 2022, Long version submitted as: the Dynamic Steiner Tree problem.
[6] M. Barjon, A. Casteigts, S. Chaumette, C. Johnen, Y.M. Neggaz, Testing temporal connectivity in sparse dynamic graphs, in: 2nd AETOS Int. Conference on Research Challenges for Future RPAS/UAV Systems, 2014.
[7] Matthieu Barjon, Arnaud Casteigts, Serge Chaumette, Colette Johnen, Yessin M. Neggaz, Testing temporal connectivity in sparse dynamic graphs, arXiv preprint, arXiv:1404.7634, 2014.
[8] Sandeep Bhadra, Afonso Ferreira, Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks, in: International Conference on Ad-Hoc Networks and Wireless, Springer, 2003, pp. 259–270.
[9] Sandeep Bhadra, Afonso Ferreira, Computing multicast trees in dynamic networks and the complexity of connected components in evolving graphs, J. Internet Serv. Appl. 3 (3) (2012) 269–275.
[10] Peter Bradshaw, Bojan Mohar, A rainbow connectivity threshold for random graph families, in: Jaroslav Nešetřil, Guillem Perarnau, Juanjo Rué, Oriol Serra (Eds.), Extended Abstracts EuroComb 2021, in: Trends in Mathematics, Springer International Publishing, 2021, pp. 842–847.

[11] Filippo Brunelli, Laurent Viennot, Minimum-cost temporal walks under waiting-time constraints in linear time, Hal preprint, https://inria.hal.science/hal-03864725v2, 2023.
[12] Arnaud Casteigts, A Journey Through Dynamic Networks (with Excursions): Habilitation à Diriger des Recherches, Université de Bordeaux, June 2018.
[13] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, Nicola Santoro, Time-varying graphs and dynamic networks, Int. J. Parallel Emerg. Distrib. Syst. 27 (5) (2012) 387–408.
[14] Arnaud Casteigts, Ralf Klasing, Yessin M. Neggaz, Joseph G. Peters, Computing parameters of sequence-based dynamic graphs, Theory Comput. Syst. 63 (3) (2019) 394–417.
[15] Arnaud Casteigts, Joseph G. Peters, Jason Schoeters, Temporal cliques admit sparse spanners, J. Comput. Syst. Sci. 121 (2021) 1–17.
[16] Lily Chen, Xueliang Li, Yongtang Shi, The complexity of determining the rainbow vertex-connection of a graph, Theor. Comput. Sci. 412 (35) (2011) 4531–4535.
[17] Julia Chuzhoy, Sanjeev Khanna, A new algorithm for decremental single-source shortest paths with applications to vertex-capacitated flow and cut problems, in: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Association for Computing Machinery, 2019, pp. 389–400.
[18] Isnard Lopes Costa, Raul Lopes, Andrea Marino, Ana Silva, On computing large temporal (unilateral) connected components, in: International Workshop on Combinatorial Algorithms, Springer, 2023, pp. 282–293.
[19] Pierluigi Crescenzi, Clémence Magnien, Andrea Marino, Approximating the temporal neighbourhood function of large temporal graphs, Algorithms 12 (10) (2019) 211.
[20] Argyrios Deligkas, Eduard Eiben, George Skretas, Minimizing reachability times on temporal graphs via shifting labels, arXiv preprint, arXiv:2112.08797, 2021.
[21] Reinhard Diestel, Graph Theory, Graduate Texts in Mathematics, vol. 173, Springer, Berlin, Heidelberg, 2017.
[22] Antoine Dutot, Frédéric Guinand, Damien Olivier, Yoann Pigné, Graphstream: a tool for bridging the gap between complex systems and dynamic graphs, in: Emergent Properties in Natural and Artificial Complex Systems. (ECCS'2007), 2007.
[23] Jessica Enright, Kitty Meeks, Hendrik Molter, Counting temporal paths, arXiv preprint, arXiv:2202.12055, 2022.
[24] Lisa Fleischer, Éva Tardos, Efficient continuous-time dynamic network flow algorithms, Oper. Res. Lett. 23 (3–5) (1998) 71–80.
[25] Satoshi Fujita, Stephane Perennes, Joseph G. Peters, Neighbourhood gossiping in hypercubes, Parallel Process. Lett. 8 (02) (1998) 189–195.
[26] Frits Göbel, J. Orestes Cerdeira, Hendrik Jan Veldman, Label-connected graphs and the gossip problem, Discrete Math. 87 (1) (1991) 29–40.
[27] Carlos Gómez-Calzado, Arnaud Casteigts, Alberto Lafuente, Mikel Larrea, A connectivity model for agreement in dynamic systems, in: Jesper Larsson Träff, Sascha Hunold, Francesco Versaci (Eds.), Euro-Par 2015: Parallel Processing, in: Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2015, pp. 333–345.
[28] John P. Hayes, A graph model for fault-tolerant computing systems, IEEE Trans. Comput. 25 (09) (1976) 875–884.
[29] Monika Henzinger, The state of the art in dynamic graph algorithms, in: 44th Intl. Conf. on Current Trends in Theory and Practice of Computer Science, SOFSEM '18, 2018, pp. 40–44.
[30] Petter Holme, Network reachability of real-world contact sequences, Phys. Rev. E 71 (4) (2005) 046119.
[31] Petter Holme, Modern temporal network theory: a colloquium, Eur. Phys. J. B 88 (9) (2015) 234.
[32] Charles Huyghues-Despointes, Binh-Minh Bui-Xuan, Clémence Magnien, Forte delta-connexité dans les flots de liens, in: ALGOTEL 2016-18èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications, 2016.
[33] David Ilcinkas, Ralf Klasing, Ahmed Mouhamadou Wade, Exploration of constantly connected dynamic graphs based on cactuses, in: International Colloquium on Structural Information and Communication Complexity, Springer, 2014, pp. 250–262.
[34] Amol Kapoor, Xue Ben, Luyang Liu, Bryan Perozzi, Matt Barnes, Martin Blais, Shawn O'Banion, Examining covid-19 forecasting using spatio-temporal graph neural networks, arXiv preprint, arXiv:2007.03113, 2020.
[35] Gueorgi Kossinets, Jon Kleinberg, Duncan Watts, The structure of information pathways in a social communication network, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, Association for Computing Machinery, August 2008, pp. 435–443.
[36] Vassilis Kostakos, Temporal graphs, Physica A 388 (6) (2009) 1007–1023.
[37] Michael Krivelevich, Raphael Yuster, The rainbow connection of a graph is (at most) reciprocal to its minimum degree, J. Graph Theory 63 (3) (2010) 185–191.
[38] David C. Kutner, Laura Larios-Jones, Temporal reachability dominating sets: contagion in temporal graphs, arXiv:2306.06999, 2024.
[39] Matthieu Latapy, Tiphaine Viard, Clémence Magnien, Stream graphs and link streams for the modeling of interactions over time, Soc. Netw. Anal. Min. 8 (1) (2018) 61.
[40] George B. Mertzios, Hendrik Molter, Rolf Niedermeier, Viktor Zamaraev, Philipp Zschoche, Computing maximum matchings in temporal graphs, arXiv preprint, arXiv:1905.05304, 2019.
[41] John W. Moon, Leo Moser, On cliques in graphs, Isr. J. Math. 3 (1) (1965) 23–28.
[42] Mohammed Yessin Neggaz, Automatic classification of dynamic graphs, PhD thesis, 2016, Thèse de doctorat dirigée par Johnen, ColetteChaumette, Serge et Casteigts, Arnaud Informatique Bordeaux 2016, http://www.theses.fr/2016BORD0169.
[43] Vincenzo Nicosia, John Tang, Mirco Musolesi, Giovanni Russo, Cecilia Mascolo, Vito Latora, Components in time-varying graphs, Chaos 22 (2) (2012) 023101.
[44] Nicholas Rescher, Alasdair Urquhart, Temporal Logic, vol. 3, Springer Science & Business Media, 2012.
[45] John M. Robson, Finding a maximum independent set in time o (2n/4), Technical report, Technical Report 1251-01, LaBRI, Université Bordeaux I, 2001.
[46] Mathilde Vernet, Yoann Pigné, Eric Sanlaville, A study of connectivity on dynamic graphs: computing persistent connected components, 4OR 21 (2) (2023) 205–233.
[47] Dong Wen, Lu Qin, Ying Zhang, Lijun Chang, Ling Chen, Enumerating k-vertex connected components in large graphs, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), April 2019, pp. 52–63.
[48] B. Bui Xuan, Afonso Ferreira, Aubin Jarry, Computing shortest, fastest, and foremost journeys in dynamic networks, Int. J. Found. Comput. Sci. 14 (02) (2003) 267–285.
[49] Ping Yu, Zhiping Wang, Peiwen Wang, Haofei Yin, Jia Wang, Dynamic evolution of shipping network based on hypergraph, Physica A 598 (July 2022) 127247.
[50] Wenyu Zang, Peng Zhang, Chuan Zhou, Li Guo, Discovering multiple diffusion source nodes in social networks, Proc. Comput. Sci. 29 (2014) 443–452.
[51] Philipp Zschoche, Till Fluschnik, Hendrik Molter, Rolf Niedermeier, The complexity of finding small separators in temporal graphs, J. Comput. Syst. Sci. 107 (2020) 72–92.