Technische Universität Berlin

Fakultät IV
Institute of Software Engineering and Theoretical Computer Science
Research Group Algorithmics and Computational Complexity

On Finding Separators in Temporal Graphs

Master Thesis

Philipp Zschoche

September 28, 2017

Supervisor and first reviewer:	Prof. Dr. Rolf Niedermeier
Second reviewer:	Prof. Dr. Martin Skutella
Co-Supervisors:	Till Fluschnik and Hendrik Molter

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Die selbstständige und eigenhändige Ausfertigung versichert an Eides statt

Berlin, _____
Ort, Datum

_____
Unterschrift

Zusammenfassung

In dieser Arbeit studieren wir die parametrisierte Komplexität der Separierung zweier Knoten in einem temporalen Graphen. Ein temporaler Graph besteht aus einer Knotenmenge V und einer Zeitkantenmenge E. Letzters sind binäre Wechselwirkungen zwischen Knoten, die mit einem Zeitstempel beschriftet sind. Es gibt verschiedene Möglichkeiten das Verhalten von Pfaden in temporalen Graphen zu modellieren. Wir fokussieren uns auf die Modelle von strikten und nicht-strikten Pfaden. Ein nicht-strikter (s, z)-Pfad ist ein Pfad vom Knoten s zum Knoten z, sodass für alle Beschriftungen t_1, t_2 von zwei aufeinanderfolgenden Kanten auf einem Pfad stets $t_1 \leqslant t_2$ gilt. Ein strikter (s, z)-Pfad ist ein nicht-strikter (s, z)-Pfad mit der zusätzlichen Einschränkung, dass die Ungleichung zwischen t_1 und t_2 strikt ist ($t_1 < t_2$). Das Problem der Nicht-Strikten (s, z)-Separation (Strikten (s, z)-Separation) fragt, ob es eine Menge S von höchstens k Knoten gibt, sodass der temporale Graph ohne S keinen nicht-strikten (s, z)-Pfad (strikten (s, z)-Pfad) hat.

Wir zeigen für die Parameter $|V|$, $k + \tau$, und $\tau + \mathrm{tw}_{\downarrow}$, dass Strikte (s, z)-Separation „fixed-parameter tractable“ ist, wobei τ die maximale Beschriftung und $\mathrm{tw}_{\downarrow}$ die Baumweite des unbeschrifteten Graphen ist. Des Weiteren zeigen wir für die Parameter $|V|$, $\tau+\mathrm{tw}_{\downarrow}$, $k+\tau+\mathrm{tw}_{\max}$, dass Nicht-Strikte (s, z)-Separation „fixed-parameter tractable“ ist, wobei $\mathrm{tw}_{\max}$ die maximale Baumweite des temporalen Graphen zu einem beliebigen Zeitpunkt ist. Falls $\mathsf{NP} \not\subseteq \mathsf{coNP}/poly$ ist, so hat (Nicht-)Strikte (s, z)-Separation für die den Parameter $k+\tau+\mathrm{tw}_{\downarrow}+\mathrm{tw}_{\max}+\Delta$ jedoch keinen polynomiellen Kern, wobei Δ der maximale Knotengrad des unbeschrifteten Graphen ist.

Andererseits zeigen wir, dass (Nicht-)Strikte (s, z)-Separation für die Parameter k, τ, $\mathrm{tw}_{\max}$, $k+\mathrm{tw}_{\max}$, und $\tau+\mathrm{tw}_{\max}$ (vermutlich) nicht „fixed-parameter tractable“ ist. Ein Unterschied ist, dass Nicht-Strikte (s, z)-Separation bereits bei $\tau = 2$ NP-schwer ist, aber Strikte (s, z)-Separation noch bis $\tau = 4$ in polynomieller Zeit lösbar ist und erst bei $\tau \geqslant 5$ NP-schwer wird.

Allerdings ist Strikte (s, z)-Separation auf planaren temporalen Graphen für den Parameter τ „fixed-parameter tractable“. Ein temporaler Graph ist planar, wenn der unbeschriftete Graph planar ist. Im Bezug auf exakte Lösungen ist ein „fixed-parameter tractable“ Resultat für (Nicht-)Strikte (s, z)-Separation auf planaren temporalen Graphen das Beste auf was wir (vermutlich) hoffen können, denn wir konnten zeigen, dass (Nicht-)Strikte (s, z)-Separation auf planaren temporalen Graphen NP-schwer ist. Hier beantworten wir eine bislang offene Frage von Fluschnik et al. [ICALP 2016] indem wir zeigen, dass Length-Bounded (s, z)-Separation auf planaren Graphen NP-schwer ist.

Anschließend betrachten wir β-Gebunde (s, z)-Separation. Dieses Problem ist dasselbe wie Nicht-Strikte (s, z)-Separation mit der zusätzlichen Einschränkung, dass die Differenz der Beschriftungen von zwei in einem Pfad aufeinanderfolgenden Kanten höchstens β ist. Wir beweisen, dass β-Gebunde (s, z)-Separation für den Parameter k W[1]-schwer ist und für den Parameter $|V|$ „fixed-parameter tractable“ ist. Jedoch hat β-Gebunde (s, z)-Separation für den Parameter $|V|$ keinen polynomiellen Kern, falls $\mathsf{NP} \not\subseteq \mathsf{coNP}/poly$.

Abstract

In this work we study the parameterized complexity of separating two vertices in a temporal graph. A temporal graph consists of a set V of vertices and a set E of time-edges, i.e. binary interactions between the vertices, labeled with time-stamps. There are several ways how the notion of a path in temporal graphs can be modeled. We focus primarily on non-strict and strict path models. A non-strict (s, z)-path is a path from a vertex s to a vertex z such that for all labels t_1, t_2 of two consecutive edges on the path it holds that $t_1 \leqslant t_2$. A strict (s, z)-path is a non-strict (s, z)-path with the additional restriction that the inequality between t_1 and t_2 is strict ($t_1 < t_2$). The Non-Strict (s, z)-Separation (Strict (s, z)-Separation) problem asks whether there is a set S of at most k vertices such that the temporal graph without S does not have a non-strict (s, z)-path (strict (s, z)-path).

We present fixed-parameter tractability results for Strict (s, z)-Separation when parameterized by $|V|$, $k + \tau$, and $\tau + \mathrm{tw}_{\downarrow}$, where τ is the maximum label and $\mathrm{tw}_{\downarrow}$ is the treewidth of the unlabeled graph. Moreover, we present fixed-parameter tractability results for Non-Strict (s, z)-Separation when parameterized by $|V|$, $\tau + \mathrm{tw}_{\downarrow}$, and $k + \tau + \mathrm{tw}_{\max}$, where $\mathrm{tw}_{\max}$ is the maximum treewidth of the temporal graph at any point in time. Unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$, however, (Non-)Strict (s, z)-Separation does not admit a polynomial kernel when parameterized by $k + \tau + \mathrm{tw}_{\downarrow} + \mathrm{tw}_{\max} + \Delta$, where Δ is the maximum degree of the unlabeled graph.

On the contrary, we show that (Non-)Strict (s, z)-Separation is (presumably) fixed-parameter intractable when parameterized by k, τ, $\mathrm{tw}_{\max}$, $k + \mathrm{tw}_{\max}$, and $\tau + \mathrm{tw}_{\max}$. Here, one difference between Non-Strict (s, z)-Separation and Strict (s, z)-Separation is that Non-Strict (s, z)-Separation is NP-hard even if the maximum time label $\tau = 2$ while Strict (s, z)-Separation is polynomial-time solvable if $\tau \leqslant 4$ and is NP-hard even if $\tau = 5$.

However, Strict (s, z)-Separation is fixed-parameter tractable when parameterized by τ on temporal planar graphs. A temporal graph is planar if the unlabeled graph is planar. In terms of exact solutions a fixed-parameter algorithm for (Non-)Strict (s, z)-Separation on temporal planar graphs is the best we can (presumably) hope for because we showed that this problem is still NP-hard on temporal planar graphs. Here, we settle an open question by Fluschnik et al. [ICALP 2016] by showing that Length-Bounded (s, z)-Separation is NP-hard on planar graphs.

Then, we consider β-Bounded (s, z)-Separation. This problem is the same as Non-Strict (s, z)-Separation with the additional restriction that the labels of two consecutive edges of a path have an upper bound β on their difference. We prove that β-Bounded (s, z)-Separation is W[2]-hard when parameterized by k and that it is fixed-parameter tractable, but (presumably) does not admit a polynomial kernel, when parameterized by $|V|$.

Contents

Chapter 1

Introduction

Patch Tuesday, Microsoft is fixing a security vulnerability. We are a system administrator of a huge manufacturing company which uses many delicate and highly specialized software tools. It is likely that an update will break a subset of those tools. This could cause a manufacturing outage—the worst case. Therefore, we roll out updates in stages which means that first a small subset of our systems gets the update, and then we slowly update further systems until all systems are up to date. This process can take days, but if something breaks it is relatively easy to rollback the update.

The security vulnerability Microsoft is fixing affects a subset of our systems and allows an attacker to execute code remotely on a computer in its local network. Even worse, there is an affected system z which is the backbone of our manufacturing process and can only be updated during the weekend. Since the update is out, everyone has knowledge about the vulnerability and therefore, the ransomware which uses the vulnerability is already on the horizon. The time is running! We need to secure the manufacturing process.

The first stage of the update process shall update k systems because $k - 1$ of our coworkers are system administrators. In case something goes wrong, there is at least one system administrator for each system. A natural question is which systems should be in the first stage of the update process such that z cannot be infected by a malware which uses the vulnerability.

We model the situation in a graph as follows: Every system which has this vulnerability is a vertex. If two systems can communicate at time point t between Patch Tuesday and the weekend, then there is an edge between these systems with label t. Furthermore, there is a special vertex s which models the outside of our infrastructure. This can be the Internet, a device of a visitor, or other things which are able to get in touch with an affected system. If the outside can communicate with an affected system at time point t between Patch Tuesday and the weekend, then there is an edge between s and the system with label t. We refer to Figure 1.1 for an illustration.

Clearly, each path from s to z where the labels are non-decreasing from s to z represents a possibility how the outside could infect z and cause a manufacturing outage. Our goal is to have no such path from s to z in the graph after the first stage of our update process is completed.

A temporal graph is, informally speaking, a graph where the edge set may change over

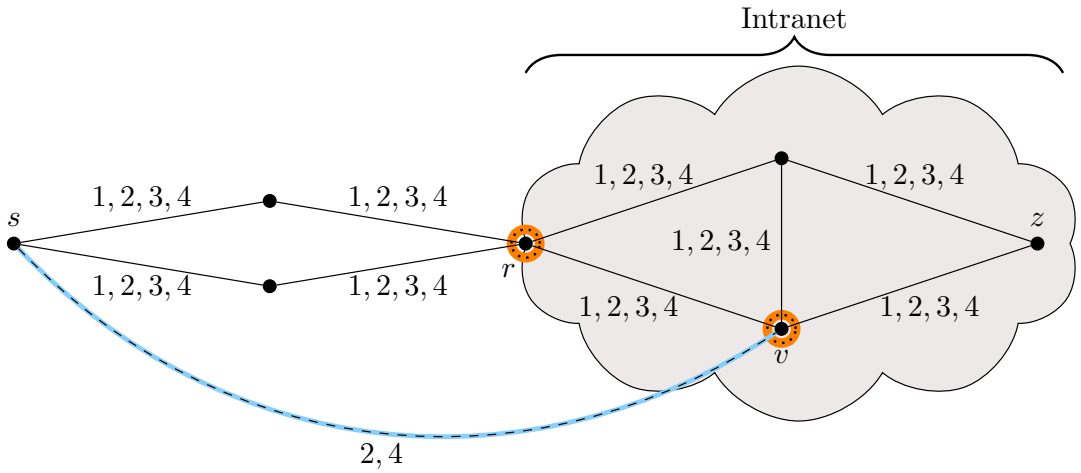


Figure 1.1: Illustration of the introductory example from Chapter 1. Each dot is an affected system. Each edge (solid line) denotes a local network connection between two systems. A label of an edge denotes on which days the connection is active. Here, "1" denotes Tuesday, "2" denotes Wednesday, "3" denotes Thursday, and "4" denotes Friday. Note that there is an intranet (gray cloud) which does not have a direct internet connection. Furthermore, an employee has meetings on Wednesday and Friday where foreign devices can access the system v. The corresponding edge is dashed (blue). If we update the systems v and r (dotted/orange), then nobody can execute code on z using the security vulnerability Microsoft is fixing.

a discrete time interval. In the literature, temporal graphs are also known as evolving [Fer04] and time-varying [Cas+12; FMS09; Nic+13] graphs, as well as temporal networks [HS12; KKK02; Mer+13], multidimensional networks [Boc+14], link streams [VLM16], and edge-scheduled networks [Ber96]. In this work, we use the well-established model in which each edge has a timestamp [Akr+15; Akr+17; Boc+14; Him+17; HS12; KKK02; Mer+13; VLM16]. Assuming discrete time steps, this is equivalent to a sequence of static graphs over a fixed set of vertices [EHK15; Mic16; MS16]. Formally, we define a temporal graph as follows.

Definition 1.1 (Temporal Graph)**.** A *temporal graph* $G = (V, E, \tau)$ is defined as an ordered triple consisting of a set of vertices V, a set of *time-edges* $E \subseteq \binom{V}{2} \times \{1, \dots, \tau\}$, and a time interval $\{1, \dots, \tau\} \subseteq \mathbb{N}$, where $\tau \in \mathbb{N}$.

See Figure 1.2 for an example. The vertices and time-edges are represented by black dots and solid lines, respectively. If there is more than one time-edge between two vertices, then there is just one solid line between these vertices, and the labels of the time-edges are comma separated.

Note that in contrast to many multilayer models, the labels have a linear order in temporal graphs and, in contrast to dynamic graphs, all layers of a temporal graph have the same vertex set.

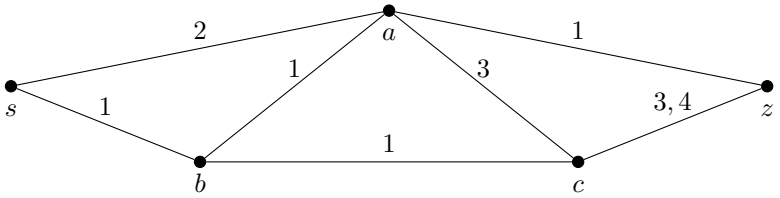


Figure 1.2: The temporal graph $G := (V, E, 4)$, where $V = \{s, a, b, c, z\}$ and $E = \{(\{s, a\}, 2), (\{s, b\}, 1), (\{b, a\}, 1), (\{b, c\}, 1), (\{a, c\}, 3), (\{a, z\}, 1), (\{c, z\}, 3), (\{c, z\}, 4)\}$.

Wu et al. [Wu+14] collected a large set of real-world scenarios which can be modeled as a temporal graph. For example, A calls B at time t in telephone network, A sends message to B at time t in a messaging networks, A follows B at time t in social networks, A cites B at time t in citation networks, A works with B at time t in collaboration networks, and information spreads from A to B at time t in information dissemination networks. A natural application on a resulting temporal graphs is whether some kind of information could spread from some vertex s to another vertex z. This work is devoted to one natural adversary question to this. We search for a set of vertices such that if the vertices are removed from the temporal graph, then information cannot spread from s to z. In our introductory example, the information was a ransomware and we removed vertices from the temporal graph by updating systems.

We mainly focus on two variants of this problem:

Non-Strict (s, z)-Separation

Input: A temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and an integer k.

Question: Is there a vertex subset $S \subseteq (V \setminus \{s, z\})$ of size at most k such that each path from s to z in G whose labels are non-strictly increasing visits at least one vertex in S?

Strict (s, z)-Separation

Input: A temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and an integer k.

Question: Is there a vertex subset $S \subseteq (V \setminus \{s, z\})$ of size at most k such that each path from s to z in G whose labels are strictly increasing visits at least one vertex in S?

Let G from Figure 1.2 and $k = 1$ be an instance of Non-Strict (s, z)-Separation or Strict (s, z)-Separation. Observe that the instance of Non-Strict (s, z)-Separation is a no-instance, because we need to remove at least two vertices but $k = 1$. However, the instance of Strict (s, z)-Separation is a yes-instance, because we can remove the vertex a or c from G to achieve the desired goal of Strict (s, z)-Separation.

Note that we refer by (Non-)Strict (s, z)-Separation to both problems Non-Strict (s, z)-Separation and Strict (s, z)-Separation. Kempe, Kleinberg, and

Kumar [KKK02] showed that (Non-)Strict (s, z)-Separation is NP-hard. A problem is NP-hard if there is presumably no exact algorithm which solves the problem in polynomial time.

In the literature, there are basically three ways how to develop efficient algorithms with provable guarantees for NP-hard problems.

First, an *approximation algorithm* is an efficient algorithm that computes a solution which is guaranteed to have at most a certain distance to an optimal solution. Second, one could restrict the input to have a specific structure and try to solve the problem on that specific structure efficiently. Third, in *parameterized complexity*, we parameterize the NP-hard problem with a parameter k and try to develop a *fixed-parameter algorithm* with respect to k. This is an algorithm which computes an optimal solution in $f(k) \cdot n^{\mathcal{O}(1)}$ time, where n is the input size and f is a computable function. Observe that the exponential blow-up of the running time is restricted to the parameter k. Hence, we have an efficient algorithm in practical applications where such a parameter is rather small. Note that assumptions on parameters are often restrictions on the input. We call a problem *fixed-parameter tractable* (in FPT) when parameterized by a parameter k if there is a fixed-parameter algorithm with respect to k for that problem.

An NP-hard problem is presumably *fixed-parameter intractable* (not fixed-parameter tractable) when parameterized by the given parameter, if the problem is W[1]-hard when parameterized by the given parameter. Observe that a problem can admit a fixed-parameter algorithm when parameterized by a specific parameter, but be W[1]-hard under another parameterization. Consequently, it is of interest to study an NP-hard problem with respect to different parameterizations.

This work is primarily devoted to study the parameterized computational complexity of (Non-)Strict (s, z)-Separation. Afterwards, we consider Non-Strict (s, z)-Separation with the additional restriction that the labels of two consecutive edges of a path have an upper bound on the distance in time to each other. We refer to Chapter 5 for a motivation and a formal definition.

Related work. Berman [Ber96] studied an analogue of the well-known theorem of Menger [Men27] in edge-scheduled networks. Note that edge-scheduled networks are essentially equivalent to temporal graphs [KKK02]. In particular, Berman [Ber96] developed an algorithm to compute the minimum number of time-edges such that the temporal graph without these time-edges does not have a path from vertex s to vertex z whose labels are non-strictly increasing. Furthermore, Berman [Ber96] showed that the maximum number of disjoint paths from a vertex s to a vertex z whose labels are non-strictly increasing is not equal to the minimum number of vertices such that the temporal graph without these vertices does not have a path from s to z whose labels are non-strictly increasing. However, Kempe, Kleinberg, and Kumar [KKK02] proved that if the temporal graph without labels excludes a fixed minor, then the maximum number of disjoint paths from a vertex s to a vertex z whose labels are non-strictly increasing is equal to the minimum number of vertices such that the temporal graph without these vertices does not have a path from s to z whose labels are non-strictly increasing. This minor the is graph of Figure 1.2 without labels.

Mertzios et al. [Mer+13] derived another analogue of the theorem of Menger [Men27]

and proved that it holds for temporal graphs. Here, one has to specify at which point in time a vertex is removed from the temporal graph such that there is no path from vertex s to vertex z whose labels are strictly increasing. Furthermore, they presented an algorithm to compute a shortest path from vertex s to vertex z whose labels are strictly increasing.

Wu et al. [Wu+14] developed, in a slightly different model, efficient algorithms to find for a source vertex s a minimum path to all other vertices. In their model of a temporal graph, a time-edge is directed and can have different incoming and outgoing labels. In temporal graphs, as well as is the model of Wu et al. [Wu+14], it is not clear what a minimum path is. Wu et al. [Wu+14] considered as a minimum path between two vertices (i) the path which contains a minimum amount of edges, (ii) the path which arrives at the earliest point in time, (iii) the path which departures at the latest point in time, and (iv) the path which has the minimum elapsed time.

Akrida et al. [Akr+17] introduced temporal flows on temporal graphs and presented an polynomial-time algorithm to compute the maximum amount of flow that can pass from a source vertex s to a sink vertex z until a given point in time. Accordingly, they showed that the maximum temporal flow is equal to the minimum number of edges such that the removal of these edges destroys every path from s to z whose labels are strictly increasing.

We close the list of related work which have a directed connection to temporal graphs by referring to the survey of Michail [Mic16] for a broad view on an algorithmic perspective on temporal graphs, and to the survey of Holme and Saramäki [HS12] for an overview of applications of temporal graph as well as related models. Among others Holme and Saramäki [HS12] describe temporal graphs in cell biology, neural and brain connections, ecological systems, infra-structural networks, physical proximity, and distributed computing.

In standard graph theory, a minimum set of vertices whose removal separates a vertex s from another vertex z can be computed in polynomial time (see Lemma 3.5). Many natural generalizations of this problem become NP-hard. Among others, (Non-)Strict (s, z)-Separation is one of these generalizations of the graph separation problem for two vertices. In the last decade, graph separation problems have become one of the most intensively studied areas of parameterized complexity, leading to various helpful techniques to design fixed-parameter algorithms. The studies include *important separators* [Mar06], *shadow removal* [MR14], *treewidth reduction* [MOR13], *LP- and CSP-branching* [Cyg+13; IWY16], *randomized contractions* [Chi+16], and *homogeneous path systems* [Gui11].

Structure of the work. In the remaining part of Chapter 1 we give an overview of this thesis and introduce formal definitions from graph theory and parameterized complexity.

Chapter 2 is about the notion of temporal graphs. We state formal definitions of temporal graphs and motivate different notions of paths, cuts, and separators in temporal graphs followed by basic observations of separators in temporal graphs.

Chapter 3 discusses the computational complexity of (Non-)Strict (s, z)-Separation with respect to the parameters solution size and maximum label.

In Chapter 4, we study the (Non-)Strict (s, z)-Separation on restricted classes of temporal graphs. First, we discuss how one can lift the concept of treewidth from standard graphs to temporal graphs and investigate the computational complexity of (Non-)Strict (s, z)-Separation on classes of temporal graphs which have bounded underlying treewidth. The underlying treewidth is the treewidth of the unlabeled graph. Afterwards, we transfer a framework of Mans and Mathieson [MM14] to show fixed-parameter tractability from dynamic graphs to temporal graphs with respect to the maximum treewidth over all layers of a temporal graph. A layer of a temporal graph is the temporal graph at a specific point in time. Second, we study the computational complexity of (Non-)Strict (s, z)-Separation on temporal planar graphs. Third, we show that for almost all fixed-parameter tractability results we establish in this thesis, we (presumably) do not have polynomial kernels.

In Chapter 5, we generalize the non-strict path model such that we can set an upper bound on the time between the labels of two consecutive edges of a path and show that the corresponding separation for two vertices (presumably) does not admit a polynomial kernel when parameterized by the number of vertices in the temporal graphs.

Finally, in Chapter 6, we recap the thesis and survey emerging future research opportunities.

1.1 Our Contributions

In this thesis, we obtain the following results for (Non-)Strict (s, z)-Separation.

In Section 3.1, we show that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size k (Theorem 3.1 and Corollary 3.2).

In Section 3.2, we prove that Non-Strict (s, z)-Separation is NP-hard if the maximum label $\tau \geqslant 2$ (Theorem 3.4), but is polynomial-time solvable if the maximum label $\tau = 1$ (Observation 3.6). Next, we show that Strict (s, z)-Separation is NP-hard if the maximum label $\tau \geqslant 5$ (Theorem 3.4), but is polynomial-time solvable if the maximum label $\tau \leqslant 4$ (Observations 3.7 and 3.8, Theorem 3.13, and Proposition 3.14). Here, we prove that the Single-Source Shortest Strict Paths problem on temporal graphs can be solved in $\mathcal{O}(|V| \cdot \tau + |E|)$ time (Lemma 3.12).

In Section 3.3, we show that Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ (Theorem 3.15), as well as that (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the number $|V|$ of vertices (Corollary 3.16). Essential for the upper bound on the running time of the latter algorithm is the upper bound of the length of a (non-)strict (s, z)-path by $|V|$, as well as that $|V|$ is also an upper bound on the solution size k. Hence, (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by a parameter ρ if we can upper bound the solution size k and the length of a (non-)strict (s, z)-path by ρ. For example the parameter $k + \tau + |V_c|$, where k is the solution size, τ the maximum label, and $|V_c|$ is the maximum number of vertices in a connected component over all layers (Corollary 3.18).

In Section 4.1.1, we prove that (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the maximum label τ and the underlying treewidth. Hence, (Non-)Strict (s, z)-Separation on classes of temporal graphs of bounded

underlying treewidth is fixed-parameter tractable when parameterized by the maximum label τ.

In Section 4.1.2, we transfer a framework of Mans and Mathieson [MM14] from dynamic graphs to temporal graphs to show fixed-parameter tractability by expressing the problem in monadic second-order logic. We show that Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k, the maximum label τ, and the layer treewidth $\mathrm{tw}_{\max}$ (Theorem 4.17). The latter parameter is the maximum treewidth of the graph over all points in time. Note that this implies that (Non-)Strict (s, z)-Separation on classes of temporal graphs of bounded layer treewidth is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ (Theorems 3.15 and 4.17). Afterwards, we prove that neither the parameter maximum label τ nor the parameter solution size k can (presumably) be dropped from the latter algorithm. In particular, we show NP-hardness of (Non-)Strict (s, z)-Separation for constant layer treewidth and maximum label (Corollary 4.18), as well as W[1]-hardness of (Non-)Strict (s, z)-Separation when parameterized by the solution size k, even if the layer treewidth is one (Corollary 4.19).

In Section 4.2, we define the class of temporal planar graphs and show that (Non-) Strict (s, z)-Separation is NP-hard (Lemma 4.20), as well as that Strict (s, z)-Separation is in FPT when parameterized by the maximum label τ on this class (Corollary 4.26). Here, we get that Strict (s, z)-Separation on classes of temporal graphs of local bounded underlying treewidth is in FPT when parameterized by the maximum label τ (Corollary 4.25) and settle an open question of Fluschnik et al. [Flu+16] whether the Length-Bounded (s, z)-Separation is NP-hard on planar graphs (Theorem 4.23).

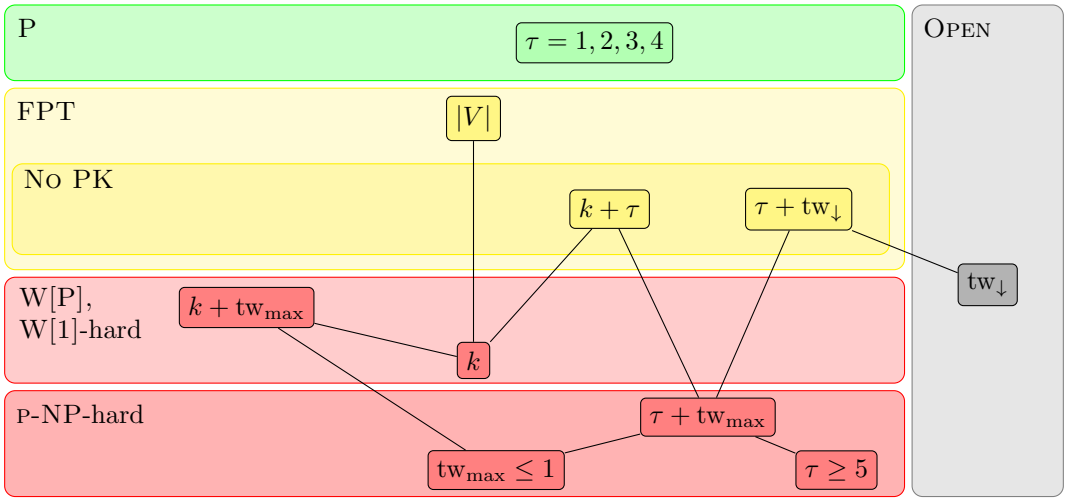
In Section 4.3, we show that unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$, (Non-)Strict (s, z)-Separation does not admit a polynomial kernel when parameterized by $k + \tau + \mathrm{tw}_{\downarrow} + \Delta$ (Theorem 4.32), where

- k is the solution size,
- τ is the maximum label,
- $\mathrm{tw}_{\downarrow}$ is the treewidth of the underlying graph, and
- Δ is the maximum degree in the temporal graph.

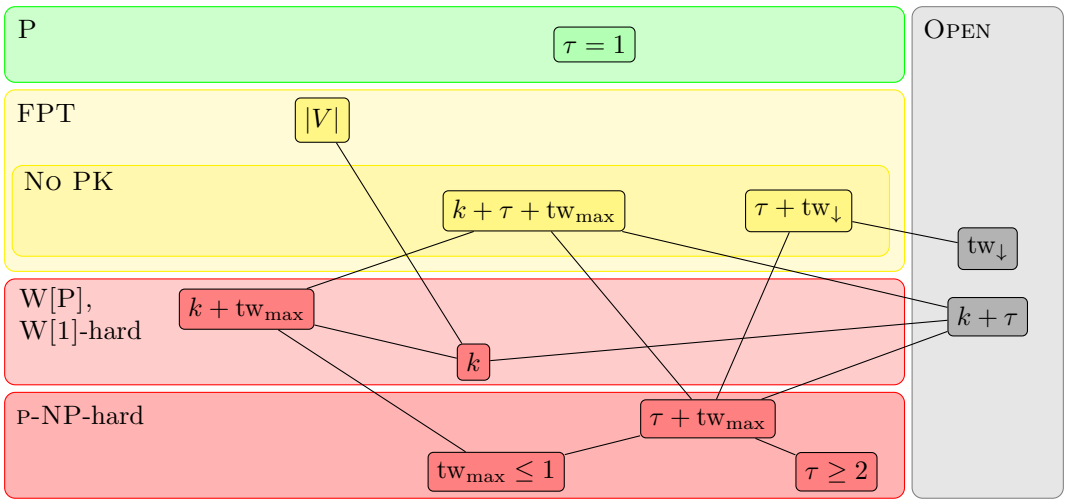
We claim that one can show, with the tool from Theorem 4.23 that Theorem 4.32 also holds on temporal planar graphs. Observe that this covers all parameters we have considered for (Non-)Strict (s, z)-Separation except the number $|V|$ of vertices and the maximum number $|V_c|$ of vertices in a connected component over all layers.

We refer to Figure 1.3a and Figure 1.3b for an overview of the parameterized complexity of Strict (s, z)-Separation and Non-Strict (s, z)-Separation, respectively. Furthermore, we refer to Table 1.1 for an survey of the results of this work with respect to (Non-)Strict (s, z)-Separation.

In Chapter 5, we generalize the non-strict path model such that we can set an upper bound on the time between the labels of two consecutive edges on a path. The corresponding separation problem for two vertices is called β-Bounded (s, z)-Separation. We show that β-Bounded (s, z)-Separation is W[2]-hard when parameterized by the solution size k (Theorem 5.1) and is fixed-parameter tractable (Corollary 5.5), but (presumably) does not admit a polynomial kernel (Theorem 5.6) when parameterized by the number $|V|$ of vertices in the temporal graph.



(a) Strict (s, z)-Separation



(b) Non-Strict (s, z)-Separation

Figure 1.3: Overview for the parameterized complexity of (Non-)Strict (s, z)-Separation. (Non-)Strict (s, z)-Separation is in P, is in FPT, is in W[P], is W[1]-hard, is p-NP-hard, is open, or has (presumably) no polynomial kernel when parameterized by $\rho \subseteq \{k, \tau, \mathrm{tw}_{\downarrow}, \mathrm{tw}_{\max}, |V|\}$ if ρ is placed in the rectangle of the mentioned property. Hence, k is the solution size, τ is the maximum label, $\mathrm{tw}_{\downarrow}$ is the underlying treewidth, $\mathrm{tw}_{\max}$ is the layer treewidth, and $|V|$ is the number of vertices. A problem is in Pif it has a polynomial-time algorithm. A problem is p-NP-hard if it is NP-hard even if the parameter is a constant. We refer to Section 1.2 for a definition of W[P]. No PK stands for "no polynomial kernel under the assumption that NP $\not\subseteq$ coNP/poly". There is an edge between two parameters if the lower parameter can be upper bounded by the upper parameter. References for the results can be found in Table 1.1.

Table 1.1: Survey of the most important results of this work for the (Non-)Strict (s,z)-Separation with respect to the parameters number $|V|$ of vertices, size k of the separator, maximum time-edge label τ, layer treewith $\mathrm{tw}_{\max}$, underlying treewidth $\mathrm{tw}_{\downarrow}$, and maximum number $|V_c|$ of vertices in a connected component over all layers. "*No PK*" stands for "no polynomial kernel under the assumption that NP $\not\subseteq$ coNP/poly". "*open*" denotes that the parameterized complexity is unknown. "*PK?*" denotes that it is open whether a polynomial kernel exists. "MSO-FPT" denotes that the FPT result is due to a monadic second-order logic formula and a theorem of Mans and Mathieson [MM14].

	(s,z)-Separation	
Parameter	Strict Paths	Non-Strict Paths
k	W[1]-h [Thm 3.1], W[P] [Obs 3.3]	W[1]-h [Cor 3.2], W[P] [Obs 3.3]
τ $\tau=1$	$\mathcal{O}(\lvert E\rvert)$ [Obs 3.7]	$\mathcal{O}(k\cdot(\lvert V\rvert+\lvert E\rvert))$ [Obs 3.6]
$\tau=2$	$\mathcal{O}(k\cdot(\lvert V\rvert+\lvert E\rvert))$ [Obs 3.8]	NP-h [Thm 3.4]
$\tau=3$	$\mathcal{O}(k\cdot\lvert V\rvert\cdot\lvert E\rvert)$ [Prop 3.14]	NP-h [Thm 3.4]
$\tau=4$	$\mathcal{O}(\lvert E\rvert\cdot\lvert V\rvert^2)$ [Thm 3.13]	NP-h [Thm 3.4]
$\tau\geqslant 5$	NP-h [Thm 3.4]	NP-h [Thm 3.4]
$\tau+k$	$\mathcal{O}(\tau^{k+3}\cdot\lvert V\rvert+\lvert E\rvert)$ [Thm 3.15], No PK [Thm 4.32]	*open*, No PK [Thm 4.32]
$\tau+k+\lvert V_c\rvert$	$\mathcal{O}(\tau^{k+3}\cdot\lvert V\rvert+\lvert E\rvert)$ [Thm 3.15], *PK?*	$\mathcal{O}((\tau\cdot\lvert V_c\rvert)^{k+3}\cdot\lvert V\rvert+\lvert E\rvert)$ [Cor 3.18], *PK?*
$\tau+\mathrm{tw}_{\max}$	NP-h [Cor 4.18], even if $\tau=6,\mathrm{tw}_{\max}=1$	NP-h[Cor 4.18], even if $\tau=4,\mathrm{tw}_{\max}=1$
$\tau+\mathrm{tw}_{\max}+k$	$\mathcal{O}(\tau^{k+3}\cdot\lvert V\rvert+\lvert E\rvert)$ [Thm 3.15], No PK [Thm 4.32]	MSO-FPT [Thm 4.17], No PK [Thm 4.32]
$\tau+\mathrm{tw}_{\downarrow}$	FPT [Thm 4.6], No PK [Thm 4.32]	FPT [Thm 4.6], No PK [Thm 4.32]
$\mathrm{tw}_{\max}+k$	W[1]-hard [Cor 4.19]	W[1]-hard [Cor 4.19]
$\lvert V\rvert$	$\mathcal{O}(\lvert V\rvert^{\lvert V\rvert+1}\cdot\tau+\lvert E\rvert)$ [Cor 3.17], *PK?*	$\mathcal{O}(\lvert V\rvert^{\lvert V\rvert+1}\cdot\tau+\lvert E\rvert)$ [Cor 3.17], *PK?*
Temporal Planar Graphs (Local Bounded Underlying Treewidth)		
	NP-h [Thm 4.23]	NP-h [Thm 4.23]
τ	FPT [Cor 4.26], *PK?*	*open*, *PK?*

1.2 Preliminaries

In this section, we present formal definitions from (standard) graph theory and parameterized complexity.

As a convention, by $\mathbb{N}$ we denote the natural numbers without zero.

Graph Theory. Let $G = (V, E)$ be an *(undirected) graph*. The set V of *vertices* is also denoted by $V(G)$ and the set E of *edges* by $E(G)$. Usually, we assume that G has no loops and hence for all $\{v, w\} \in E(G)$ it holds that $v \neq w$. Two vertices $v, w \in V(G)$ are *adjacent* if there is an edge $\{v, w\} \in E(G)$. Two edges $e_1, e_2 \in E(G)$ are *incident* if $e_1 \cap e_2 \neq \emptyset$. A vertex $v \in V(G)$ is *incident* with an edge $e \in E(G)$ if $v \in e$. For an edge set $E' \subseteq E$, we denote by $G \setminus E'$ the deletion of edges E' in G. For a vertex set $V' \subseteq V$, we denote by $G - V'$ the deletion of V' in G with the vertex set $V(G) \setminus V'$ and the edge set $\{\{v, w\} \mid v, w \in V(G) \setminus V'\}$. The *neighborhood* of a vertex v in G is defined as $N(v) := \{w \in V(G) \mid \{v, w\} \in E(G)\}$ and the *maximum degree* of G is denoted by $\Delta(G) := \max_{v \in V(G)} |N(v)|$. For a vertex set $X \subseteq V(G)$, we denote by $G[X]$ the *(induced) subgraph* $G-(V(G)\setminus X)$ of G and for an edge set $Y \subseteq E(G)$, we denote by $G[Y]$ the induced subgraph $(\{v \in V \mid \{v, w\} \in Y\}, Y)$. A *path* $P = e_1, \ldots, e_\ell$ in G of length ℓ is a sequence of ℓ edges such that $\Delta(G[P]) = 2$, $|e_i \cap e_{i+1}| = 1$ for all $i \in \{1, \ldots \ell - 1\}$, and $e_i \neq e_j$ for all $i \neq j \in \{1, \ldots, \ell\}$.
The path P is an *(s, z)-path* if it starts at vertex $s = e_1 \setminus e_2$ and ends in $z = e_\ell \setminus e_{\ell-1}$ or if P contains only one edge and $\{s, z\} = e_1$. A *cycle* is an (s, s)-path. The vertices visited by a path $P = e_1, \ldots, e_\ell$ are denoted by $V(P) = \bigcup_{i=1}^{\ell} e_i$.

Two (s, z)-paths P_1 and P_2 in a graph are *vertex-disjoint* if $(V(P_1) \setminus \{s, z\}) \cap (V(P_2) \setminus \{s, z\}) = \emptyset$ and *edge-disjoint* if P_1 and P_2 do not have an edge in common.

We say that the subgraph $G[X]$ is a *connected component* (or *connected*) if for each vertex pair $v \neq w \in X$ there is a (v, w)-path in $G[X]$. The *maximum connected component size* of G is the maximum size of a subset $V' \subseteq V(G)$ such that $G[V']$ is a connected component.

A graph $G = (V_1, V_2, E)$ is *bipartite* if $(V_1 \cup V_2, E)$ is a graph, $V_1 \cap V_2 = \emptyset$, and $G[V_1]$ and $G[V_2]$ are edgeless.

A *tree* is a connected graph which does not contain a cycle.

An $n \times n$ *grid* is a graph $G = (V, E)$ such that $V := \{(i, j) \mid i, j \in \{1, \ldots, n\}\}$ and $E := \{\{(i, j), (i', j')\} \mid |i - i'| + |j - j'| = 1\}$.

A *directed graph* $G = (V, E)$ consists of a set of vertices V and a set of edges $E = \{(v, w) \mid v, w \in V\}$. Every edge $(v, w) \in E$ has a direction from the *source* v to *destination* w. A *(directed) path* $P = e_1, \ldots, e_\ell$ is a sequence of edges such that the destination of e_i is the source of e_{i+1} for all $i \in \{1, \ldots, \ell-1\}$. The directed path P is a *(directed) (s, z)-path* if the source of e_1 is s and the destination of e_ℓ is z. A directed graph $G = (V, E)$ is a *directed acyclic graph (DAG)* if G does not contain a directed (v, v)-path for all $v \in V$.
For the rest of this work, a graph G is undirected of not stated otherwise. For more information on classical graph theory, we refer to Diestel [Die16].

Parameterized Complexity. Parameterized complexity is a branch of computational complexity theory that focuses on classifying computational problems according to their

inherent difficulty with respect to a parameter. A *problem* is a formal language $L \subseteq \Sigma^*$, where Σ is a finite alphabet. An instance $I \in \Sigma^*$ of problem L of length $|I|$ is a yes-instance if and only if $I \in L$, otherwise we call I a no-instance.

An instance (I, r) of a *parameterized problem* consists of the actual instance I and of an integer r referred to as the *parameter* [Cyg+15; DF13; FG06; Nie06]. A parameterized problem is called *fixed-parameter tractable* if there is a (fixed-parameter tractable) algorithm that solves each instance (I, r) in $f(r) \cdot |I|^{\mathcal{O}(1)}$ time, where f is a computable function depending only on the parameter r. For a recent survey on fixed-parameter algorithm design techniques, we refer to Cygan et al. [Cyg+15].

The class of fixed-parameter tractable problems is denoted by FPT and the class XP contains all problems where each instance (I, r) can be solved in $|I|^{f(r)}$, where f is a computable function. One can show that a parameterized problem L is (presumably) not in XP, and therefore not in FPT, by showing NP-hardness where the parameter is constant. There is a hierarchy $\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \mathsf{W[2]} \subseteq \cdots \subseteq \mathsf{W[P]} \subseteq \mathsf{XP}$ of hardness classes for parameterized problems. We refer to Downey and Fellows [DF13] and Flum and Grohe [FG06] for a proper definition of the W-hierarchy. The most important classes for us are W[1] and W[2]. One can show that a parameterized problem L is (presumably) not in FPT by devising a *parameterized reduction* from a W[1]-hard or W[2]-hard problem to L.

A *parameterized reduction* from a parameterized problem L to another parameterized problem L' is a function that acts as follows: For two computable functions f and g, given an instance (I, r) of problem L, it computes in $f(r) \cdot |I|^{\mathcal{O}(1)}$ time an instance (I', r') of problem L' so that $r' \leqslant g(r)$ and $(I, r) \in L$ if and only if $(I', r') \in L'$.

A *kernelization* is an algorithm that, given an instance (I, r) of a parameterized problem L, computes in polynomial-time an instance (I', r') of L (the kernel) such that (I, r) is a yes-instance if and only if (I', r') is a yes-instance and $|I'| + r' \leqslant f(r)$ for some computable function f only depending on r. We say that f measures the size of the kernel, and if $f \in r^{\mathcal{O}(1)}$, we say that L admits a polynomial kernel. It is clear that any (decidable) parameterized problem which has a kernelization algorithm is in FPT. Moreover, all problems in FPT have kernelization algorithms [Cai+97]; however, the kernel size then typically is exponential.

Chapter 2

Temporal Graph Theory

In this chapter, we introduce and motivate certain terms and definitions for temporal graphs, show that a (non-)strict (s, z)-path can be computed in polynomial-time, and describe basic properties about paths, cuts, and separators in temporal graphs.

Let $G = (V, E, \tau)$ be a temporal graph. Two vertices $v, w \in V$ are *adjacent* if there is an edge $(\{v, w\}, t) \in E$. Two time-edges $(e_1, t_1), (e_2, t_2) \in E$ are *incident* if $e_1 \cap e_2 \neq \emptyset$. A vertex $v \in V$ is *incident* with a time-edge $(e, t) \in E$ if $v \in e$.

The graph $G_i = (V, E_i)$ is called *layer* i of the temporal graph $G = (V, E, \tau)$ if and only if $\{v, w\} \in E_i \Leftrightarrow (\{v, w\}, i) \in E$.

The *underlying graph* $G_\downarrow$ of a temporal graph $G = (V, E, \tau)$ is defined as $G_\downarrow := (V, E_\downarrow)$, where $E_\downarrow = \{e \mid (e, t) \in E\}$. In contrast to the temporal graph G, the underlying graph $G_\downarrow$ has at most one edge between two vertices.

For a vertex set $X \subseteq V$ we define the *(induced) temporal subgraph* of X by $G[X] := (X, \{(\{v, w\}, t) \in E \mid v, w \in X\}, \tau)$. The *maximum degree* $\Delta(G)$ of a temporal graph $G = (V, E, \tau)$ is the maximum degree of the underlying graph of G.

Paths in temporal graphs. It is not entirely clear how to lift the notion of a path to temporal graphs.

Suppose that we model the connectivity of a wireless sensor network as a temporal graph and want that the way a packet can go in our wireless sensor network corresponds to a path in our temporal graph. Then, we probably favor the following definition of a path. Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$. A *non-strictly increasing* (s, z)*-path* (or *non-strict* (s, z)*-path* for short) of length ℓ in G is a sequence of time-edges

$$P = (\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), \ldots, (\{v_{\ell-1}, z\}, t_\ell),$$

where no time-edge appears twice in P, $t_i \leqslant t_{i+1}$, and $(\{s, v_1\}, t_1), (\{v_i, v_{i+1}\}, t_{i+1}), (\{v_{\ell-1}, z\}, t_\ell) \in E$ for all $i \in \{1, \ldots, \ell - 1\}$. In the literature, non-strict paths are also known as non-decreasing paths [KKK02; Wil10].

Now suppose that we model a public transport system as a temporal graph: every station and every vessel is a vertex and there is a time-edge $(\{v, w\}, t)$ in the temporal graph if

(i) vessel v stops at station w at time t, and
(ii) station w and station v are on the same interchange hub.

Then, a non-strictly increasing path might not correspond to a route a passenger could take, because it cannot leave a vessel, move to another station on the same hub, and enter a vessel again at the same point in time. Of course, one could argue that this depends on how exact the resolution of time is in our temporal graph. However, a different notation of a path might be more appropriate. A *strictly increasing* (s, z)*-path* (or *strict* (s, z)*-path* for short) of length ℓ in G is a non-strict (s, z)-path where $t_i < t_{i+1}$ for all $i \in \{1, \dots, \ell - 1\}$. In the literature, strict paths are also known as journeys [Akr+15; Akr+17; Mer+13; Mic16].

Another obvious application of temporal graphs is the spreading of an epidemic. Here, the population is the set of vertices and there is a time-edge $(\{v, w\}, t)$ if individual v and individual w have physical contact at time t. Now, we want that a path from an individual s to an individual z models how a disease could spread from s to z.

On the one hand, there are diseases, like HIV, which are not contagious to others directly after the infection and, on the other hand, there are diseases which can be handled by the immune system and therefore are not contagious to others after some amount of time. An (α, β)*-bounded* (s, z)*-path* $P = (e_1, t_1), \dots, (e_\ell, t_\ell)$ of length ℓ is a non-strict (s, z)-path of length ℓ where $\alpha \leqslant t_{i+1} - t_i \leqslant \beta$ for all $i \in \{1, \dots, \ell - 1\}$.

One can observe that in a temporal graph $G = (V, E, \tau)$ a non-strict (s, z)-path is also a $(0, \tau)$-bounded (s, z)-path (or τ-bounded (s, z)-path), a strict (s, z)-path is also a $(1, \tau)$-bounded (s, z)-path, and every strict (s, z)-path or (α, β)-bounded (s, z)-path is a non-strict (s, z)-path. Furthermore, there is a close relation to multi-layer graphs—for each $(0, 0)$-bounded (s, z)-path P_0 in G there exists an $i \leqslant \tau$ such that there is an (s, z)-path in G_i which visits the same vertices as P_0 and in the same order as P_0.

We mainly devote this work to the non-strict (s, z)-path and strict (s, z)-path model, because those models are widely used is the literature [Akr+15; Akr+17; KKK02; Mer+13; Mic16; MM14; Wil10], non-strict (s, z)-path is inclusion-wise the most general one, and the strict (s, z)-path model is the most simple model in which labels in a path are strictly increasing (that is $\alpha > 0$ for the (α, β)-bounded (s, z)-path model). This is especially interesting for parameterized algorithms because here the parameter τ of the temporal graph is an upper bound for the length of a strict (s, z)-path.

Another interesting aspect of some path models in temporal graphs is that cycles might have a special role whenever $\beta < \tau$, because it is not possible to wait at a vertex an arbitrary amount of time. But one can go a cycle to bridge the waiting time. In the literature, this aspect is studied by Akrida et al. [Akr+17] as vertex buffer times and by Pan and Saramäki [PS11] under the name of waiting time cutoffs. In Chapter 5, we study the β-bounded (s, z)-path model.

The *departure time* (*arrival time*) of a (non-)strict (s, z)-path or (α, β)-bounded (s, z)-path $P = (e_1, t_1), \dots, (e_\ell, t_\ell)$ is t_1 (t_ℓ) and the *traversal time* of P is $t_\ell - t_1$. The vertices *visited* by P are denoted by $V(P) = \bigcup_{i=1}^{\ell} e_i$.

Remark 2.1. In sharp contrast to paths in standard graphs, for all models of a path in temporal graphs from above the connectivity between vertices is not transitive. Figure 2.1 provides an example for the non-strict path model.

Next, we discuss some basic data reduction rules to simplify a temporal graph which we will use throughout the whole thesis. We denote a discrete time interval from $a \in \mathbb{N}$ to $b \in \mathbb{N}$ with $[a, b] := \{x \in \mathbb{N} \mid a \leqslant x \leqslant b\}$. If an endpoint is excluded from the set then

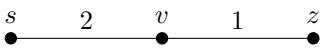


Figure 2.1: A temporal graph which has a non-strict (s, v)-path and a non-strict (v, z)-path but no non-strict (s, z)-path

we use parentheses instead of brackets. For example, $(a, b] := \{x \in \mathbb{N} \mid a < x \leqslant b\}$ if a is excluded.

Reduction Rule 2.1. *Let $G = (V, E, \tau)$ be a temporal graph and let $[t_1, t_2] \subseteq [1, \tau]$ be an interval where for all $t \in [t_1, t_2]$ the layer G_t is an edgeless graph. Then for all $(\{v, w\}, t') \in E$ where $t' > t_2$ replace $(\{v, w\}, t')$ with $(\{v, w\}, t' - t_2 + t_1 - 1)$ in E.*

The effect of Reduction Rule 2.1 on a temporal graph is illustrated in Figure 2.2. Reduction Rule 2.1 is correct and can be exhaustively applied in polynomial-time, as will be shown below in Lemma 2.1.

If Reduction Rule 2.1 is not applicable on a given temporal graph $G = (V, E, \tau)$, then there might be some edgeless layers at the end of the interval $[1, \tau]$. These layer are also removable.

Reduction Rule 2.2. *Let $G = (V, E, \tau)$ be a temporal graph. If there is a non-empty interval $[t_1, \tau]$ where for all $t' \in [t_1, \tau]$ the layer $G_{t'}$ is an edgeless graph, then set τ to $t_1 - 1$.*

Lemma 2.1. *Reduction Rules 2.1 and 2.2 do not remove or add any (non-)strict (s, z)-path from/to the temporal graph $G = (V, E, \tau)$ and can be exhaustively applied in $\mathcal{O}(|E|)$ time if E is ordered by ascending labels.*

Proof. First we discuss Reduction Rule 2.1. Let $G = (V, E, \tau)$ be a temporal graph, $s, z \in V$, $[t_\alpha, t_\beta] \subseteq [1, \tau]$ be an interval where for all $t \in [t_\alpha, t_\beta]$ the layer G_t is an edgeless graph. Let $P = (e_1, t_1), \ldots, (e_i, t_i), (e_{i+1}, t_j), \ldots, (e_n, t_n)$ be a (non-)strict (s, z)-path, and let G' be the graph after we applied Reduction Rule 2.1 once on G. We distinguish three cases.

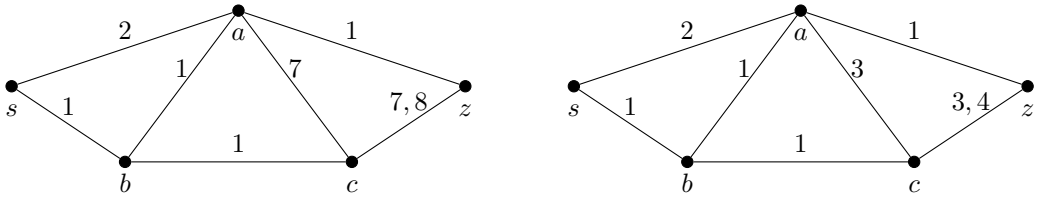
Case 1: If $t_\beta > t_n$, then no time-edge of P is touched by Reduction Rule 2.1. Hence, P also exists in G'.

Case 2: If $t_i < t_\alpha < t_\beta < t_{i+1}$, then there is a non-strict (s, z)-path $(e_1, t_1), \ldots, (e_i, t_i)$, $(e_{i+1}, t_j - t_\beta + t_\alpha - 1), \ldots, (e_n, t_n - t_\beta + t_\alpha - 1)$ in G', because $t_i < t_{i+1} - t_\beta + t_\alpha - 1$.

Case 3: If $t_\beta < t_1$, then there is clearly a non-strict (s, z)-path $(e_1, t_1 - t_\beta + t_\alpha - 1), \ldots, (e_n, t_n - t_\beta + t_\alpha - 1)$ in G'

The other direction works analogously. We look at a (non-)strict (s, z)-path in G' and compute the corresponding (non-)strict (s, z)-path in G.

Reduction Rule 2.1 can be exhaustively applied by iterating over the by time-edges (e_i, t_i) in the time-edge set E ordered by ascending labels until the first t_1, t_2 with the given requirement appear. Set $x_0 := -t_2 + t_1 - 1$. Then we iterate further over E and replace each time-edge (e, t) with $(e, t + x_0)$ until the next t_1, t_2 with the



(a) Reduction Rule 2.1 is applicable (b) Reduction Rule 2.1 is not applicable

Figure 2.2: Figure 2.2a shows a temporal graph where Reduction Rule 2.1 is applicable. In particular, layers $3, 4, 5, 6$ are edgeless. Figure 2.2b shows the same temporal graph after Reduction Rule 2.1 was applied exhaustively.

given requirement appear. Then we set $x_1 = x_0 - t_2 + t_1 - 1$ and iterate further over E and replace each time-edge (e, t) with $(e, t + x_1)$. We repeat this procedure until the end of E is reached. Since we iterate over E only once, this can be done in $\mathcal{O}(|E|)$ time.

Reduction Rule 2.2 can be executed in linear time by iterating over all edges and taking the maximum label as t_1. Note that the vertices V and the time-edges E remain untouched by Reduction Rule 2.2. Hence, the application of Reduction Rule 2.2 does not add or remove any (non-)strict (s, z)-path. □

A consequence of Lemma 2.1 is that the maximum label τ can be upper-bounded by the number of time-edges and hence the input size.

Lemma 2.2. *Let $G = (V, E, \tau)$ be a temporal graph, where Reduction Rules 2.1 and 2.2 are not applicable. Then $\tau \leqslant |E|$.*

Proof. Let $G = (V, E, \tau)$ be a temporal graph, where Reduction Rules 2.1 and 2.2 are not applicable. Then for each $t \in \{1, \ldots, \tau\}$ there is a time-edge $(e, t) \in E$. Thus,

$$\tau \leqslant \sum_{i=1}^{\tau} |\{(e, t) \in E \mid t = i\}| \leqslant |E|.$$

□

Static expansion of a temporal graph. A key tool of Akrida et al. [Akr+17], Berman [Ber96], Kempe, Kleinberg, and Kumar [KKK02], and Mertzios et al. [Mer+13] is the time-expanded version of a temporal graph which reduces reachability and other related questions in temporal graphs to similar questions in directed graphs. Let $G = (V, E, \tau)$ be a temporal graph where $V := \{v_1, \ldots, v_n\} \cup \{s, z\}$. We say that the *non-strict static expansion* of (G, s, z) is a directed graph $H := (S, A)$, where $S := \{u_{i,j} \mid 1 \leqslant i \leqslant \tau \text{ and } 1 \leqslant j \leqslant n\} \cup \{s, z\}$ and

$$A := \{(u_{(i-1),j}, u_{i,j}) \mid 1 < i \leqslant \tau\} \cup \{(u_{i,j}, u_{i,j'}), (u_{i,j'}, u_{i,j}) \mid (\{v_j, v_{j'}\}, i) \in E\} \cup \{(s, u_{i,j}) \mid (\{s, v_j\}, i) \in E\} \cup \{(u_{i,j}, z) \mid (\{v_j, z\}, i) \in E\}.$$

The *strict static expansion* of (G, s, z) is a directed acyclic graph $H' := (S, A')$ where

$$A' := \{(u_{(i-1),j}, u_{i,j}) \mid 2 < i \leqslant \tau\} \cup \{(u_{(i-1),j}, u_{i,j'}), (u_{(i-1),j'}, u_{i,j}) \mid (\{v_j, v_{j'}\}, i) \in E\} \cup \{(s, u_{i+1,j}) \mid (\{s, v_j\}, i) \in E\} \cup \{(u_{i,j}, z) \mid (\{v_j, z\}, i) \in E\}.$$

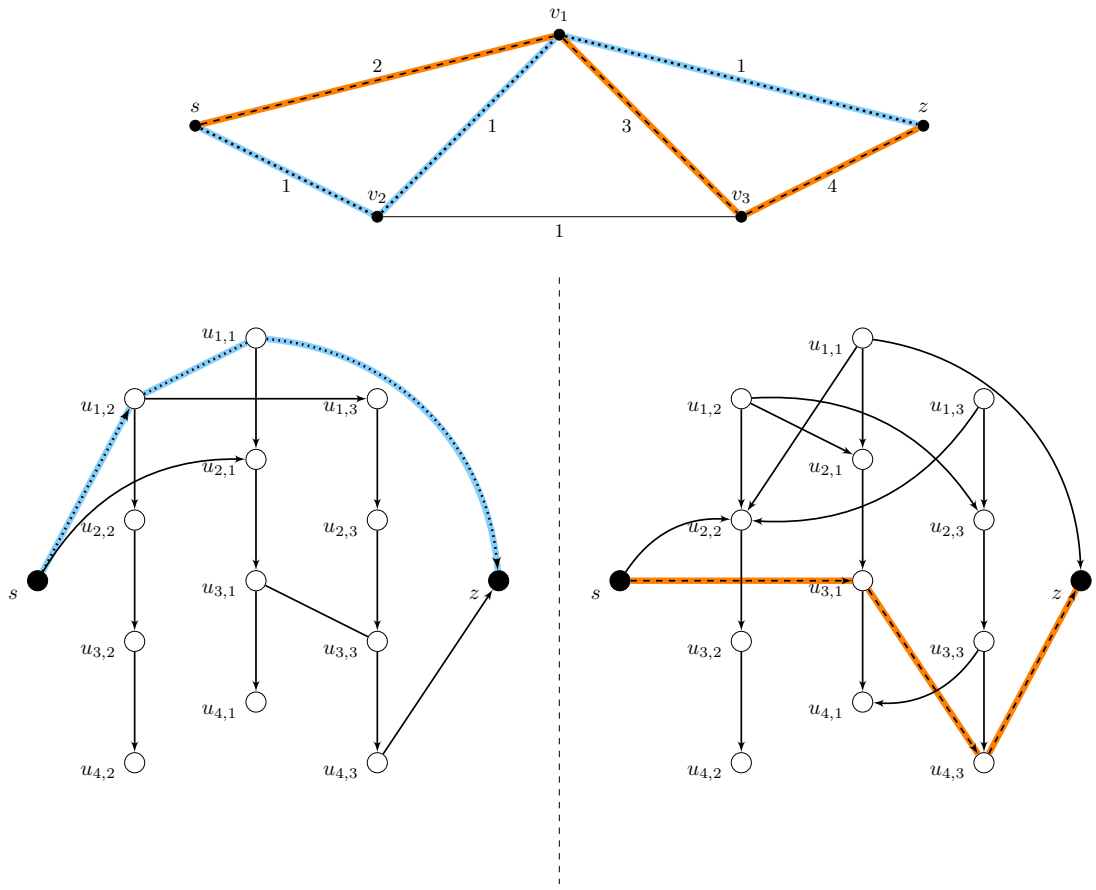


Figure 2.3: A temporal graph (top), its non-strict static expansion (left) and its strict static expansion (right). There is a non-strict (s, z)-path (blue/dotted) and a strict (s, u)-path (orange/dashed).

We say that the set $\{(u_{(i-1),j}, u_{i,j}) \mid 2 < i \leqslant \tau\} \subseteq A$ (or $\{(u_{(i-1),j}, u_{i,j}) \mid 2 < i \leqslant \tau\} \subseteq A'$) is the set of *column-edges* of H (or H'). Observe that for each non-strict (s, z)-path in G, there is a corresponding (s, z)-path in H and that for each strict (s, z)-path in G there is a corresponding (s, z)-path in H'. We refer to Figure 2.3 for an example. For algorithmic purposes, the construction time of a (non-)strict static expansion is important.

Lemma 2.3. *Let $G = (V, E)$ be a temporal graph, where $s, z \in V$ are two distinct vertices. The (non-)strict static expansion (G, s, z) can be computed in $\mathcal{O}(|V| \cdot \tau + |E|)$ time.*

Proof. Let $G = (V, E)$ be a temporal graph, where $s, z \in V$ are two distinct vertices. First, we construct the set S of vertices of the (non-)strict static expansion $H := (S, A)$ of (G, s, z) and add all column-edges to the edge set A. Since H has at most $|V| \cdot \tau$ vertices, this can be done in $\mathcal{O}(|V| \cdot \tau)$ time.

Second, we iterate once over the set E of time-edges and add the corresponding edges to A. In the case of a non-strict static expansion each time-edge has exactly one

corresponding edge in A and in the case of a strict static expansion each time-edge has at most two corresponding edges in A. Hence, the overall running time is $\mathcal{O}(|V| \cdot \tau + |E|)$. Because of Lemma 2.2, we have $\mathcal{O}(|V| \cdot \tau + |E|) \leqslant \mathcal{O}(|V| \cdot |E|)$. □

With help of the (non-)strict static expansion, one can compute a (non-)strict (s, z)-path in a temporal graph in polynomial-time.

Lemma 2.4. *Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$. A (non-)strict (s, z)-path can be computed in $\mathcal{O}(|V| \cdot \tau + |E|) \leqslant \mathcal{O}(|V| \cdot |E|)$ time.*

Proof. First, we show how a non-strict (s, z)-path can be computed and then how the algorithm can be adjusted such that we only find strict (s, z)-paths.

The algorithm for a non-strict (s, z)-path works as follows:

(i) Construct the non-strict static expansion $H = (S, A)$ of (G, s, z).

(ii) Perform a breadth-first search from s to z in H to find an (s, z)-path P.

(iii) Reconstruct a non-strict (s, z)-path P' from P by ignoring the edges from a vertex $u_{i,j}$ to vertex $u_{i+1,j}$ and take for each edge $(u_{i,j}, u_{i,j'}) \in P$ the time-edge $(\{j, j'\}, i) \in E$.

One can observe that there is a non-strict (s, z)-path in G if and only if there is an (s, z)-path in the non-strict static expansion of (G, s, z). This implies the correctness of this algorithm.

If we want to decide whether there is a strict (s, z)-path in G, then we use the same algorithm but instead of the non-strict static expansion of (G, s, z) we construct the strict static expansion of (G, s, z) in (i).

Note that $|S| \in \mathcal{O}(|V| \cdot \tau)$ and $|A| \in \mathcal{O}(|E| + \tau \cdot |V|)$. The running time of this algorithm is $\mathcal{O}(|V| \cdot \tau + (|E| + \tau \cdot |V|)) = \mathcal{O}(|V| \cdot \tau + |E|)$. Because of Lemma 2.2, we know that we can compute a temporal graph equivalent to G where $\tau \leqslant |E|$. Hence, the running time is in $\mathcal{O}(|V| \cdot |E|)$. □

Cuts and separators in temporal graphs. To be consistent with Diestel [Die16] we call path elimination problems by edge deletion *cuts* and by vertex deletion *separators*. Since we mainly focus on the non-strict (s, z)-path model and the strict (s, z)-path model, we will define cuts and separators only for those models. Later in Chapter 5, we will also define separators for the β-bounded (s, z)-path model.

Let $G = (V, E, \tau)$ be a temporal graph, $S \subseteq V$ a set of vertices, and $C \subseteq E$ a set of time-edges. We denote the temporal graph G without S by $G - S := (V \setminus S, \{(\{v, w\}, t) \in E \mid v, w \in V \setminus S\}, \tau)$ and the temporal graph G without C by $G \setminus C := (V, E \setminus C, \tau)$. Since we have two different notions of a path, we need two different kinds of cuts and separators.

- A vertex set S is a *non-strict (s, z)-separator* if there is no non-strict (s, z)-path in $G - S$ and a time-edge set C is a *non-strict (s, z)-cut* if there is no non-strict (s, z)-path in $G \setminus C$.

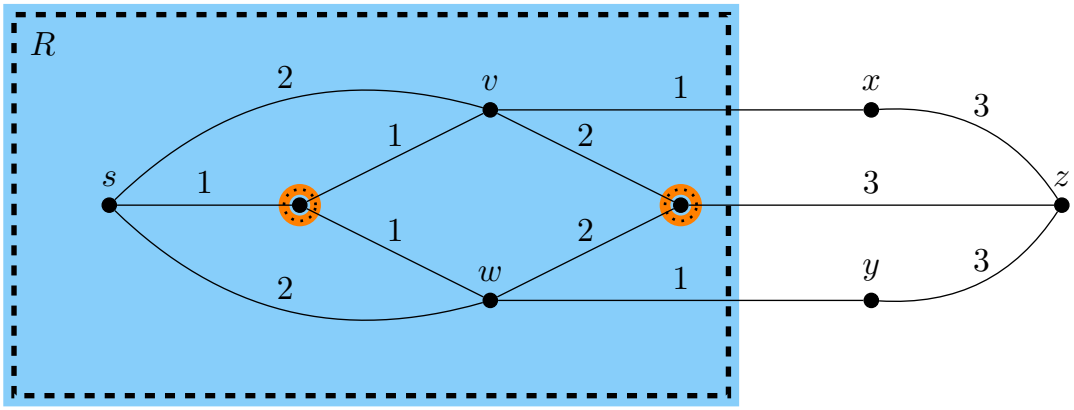


Figure 2.4: A temporal graph with a non-strict (s, z)-separator S (dotted/orange) where R is the set of vertices which are reachable from s. Note that $v, w, x, y \notin S$ and the time-edges $(\{v, x\}, 1)$ and $(\{w, y\}, 1)$ have one endpoint in R and the other endpoint not in R.

- A vertex set S is a *strict* (s, z)*-separator* if there is no strict (s, z)-path in $G - S$ and a time-edge set C is a *strict* (s, z)*-cut* if there is no strict (s, z)-path in $G \setminus C$.

Remark 2.2. In a graph $G = (V, E)$, for each (s, z)-cut C there is a set R of vertices which are reachable from s. Then, each edge in C has one endpoint in R and one endpoint in $V \setminus R$. Analogously, if we want an (s, z)-separator S, then at least one of the endpoints of an edge in C is in S. This is not the case with temporal graphs. As an example consider the temporal graph in Figure 2.4 with a non-strict (s, z)-separator (orange/dotted), a set of vertices (blue/dashed) which are reachable by a non-strict path from s, and two time-edges where one endpoint is in the reachable set, but none of the endpoints is in the non-strict (s, z)-separator.

Now, we can express the (Non-)Strict (s, z)-Separation problem in terms of our definitions.

Non-Strict (s, z)-Separation
Input: A temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and an integer k.
Question: Is there a non-strict (s, z)-separator S of size at most k?

Strict (s, z)-Separation
Input: A temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and an integer k.
Question: Is there a strict (s, z)-separator S of size at most k?

For both problems, we assume over the whole thesis that there is no time-edge between s and z because these instances are trivial no-instances.

In the remaining part of this section we discuss how the minimum size of a (non-) strict (s, z)-separator can be lower- and upper-bounded.

We say that two (non-)strict (s, z)-paths P_1 and P_2 in a graph are *vertex-disjoint* if $(V(P_1) \setminus \{s, z\}) \cap (V(P_2) \setminus \{s, z\}) = \emptyset$ and *edge-disjoint* if P_1 and P_2 do not have a time-edge in common.

The famous theorem of Menger [Men27] in graph theory uncovers one of the most important properties of a graph.

Theorem 2.5 (Menger [Men27])**.** *Let $G = (V, E)$ be a (directed) graph, $s, z \in V$ and $\{s, z\} \notin E$. Then,*

(i) the maximum number of pairwise vertex-disjoint (s, z)-paths in G is equal to the minimum size of an (s, z)-separator in G, and

(ii) the maximum number of pairwise edge-disjoint (s, z)-paths in G is equal to the minimum size of an (s, z)-cut in G.

Berman [Ber96] observed that, in temporal graphs, condition (i) of Theorem 2.5 does not hold for (non-)strict (s, z)-paths. Kempe, Kleinberg, and Kumar [KKK02] showed that condition (i) of Theorem 2.5 holds for a given temporal graph with respect to (non-)strict (s, z)-paths if its underlying graph does not have a fixed minor. As already mention that fixed minor is the underlying graph of the temporal graph of Figure 1.2.

However, observe that the maximum number of vertex-disjoint (non-)strict (s, z)-paths is a lower bound for the minimum size of a (non-)strict (s, z)-separator.

Observation 2.6. *Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$. The maximum number of vertex-disjoint (non-)strict (s, z)-paths in G is at most the minimum size of a (non-)strict (s, z)-separator in G.*

Proof. Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$.

$\Rightarrow$: Suppose that there are k vertex-disjoint (non-)strict (s, z)-paths $P_1, \ldots, P_k$ and assume towards a contradiction that there is a (non-)strict (s, z)-separator S of size $k - 1$. Since S is a (non-)strict (s, z)-separator, it holds that $|V(P_i) \cap S| \geqslant 1$ for all $i \in \{1, \ldots, k\}$. This is a contradiction because the size of S is $k - 1$ and $P_1, \ldots, P_k$ are vertex-disjoint.

$\Leftarrow$: Let S be a (non-)strict (s, z)-separator of size k and assume towards a contradiction that there are $k + 1$ vertex-disjoint (non-)strict (s, z)-paths $P_1, \ldots, P_{k+1}$. Since S is a (non-)strict (s, z)-separator, it holds that $|V(P_i) \cap S| \geqslant 1$ for all $i \in \{1, \ldots, k+1\}$. Hence, there are $i, j \in \{1, \ldots, k + 1\}$ with $i \neq j$ such that $V(P_i) \cap V(P_j) \neq \emptyset$ because there are only k vertices in S. This is a contradiction. □

Kempe, Kleinberg, and Kumar [KKK02] showed that the gap between the maximum number of vertex-disjoint (non-)strict (s, z)-paths and the minimum size of a (non-)strict (s, z)-separator can be arbitrarily large. Furthermore, the maximum number of vertex-disjoint (non-)strict (s, z)-paths in a temporal graph is NP-hard to compute.

Theorem 2.7 (Kempe, Kleinberg, and Kumar [KKK02])**.** *Let $G = (V, E, t)$ be a temporal graph and $s, z \in V$. It is NP-complete to decide whether there exist two vertex-disjoint (non-)strict (s, z)-paths.*

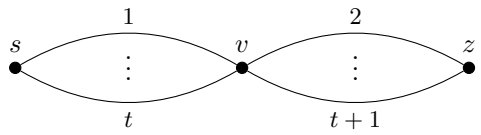


Figure 2.5: Temporal graph $G = (V, E, t + 1)$ with the (non-)strict (s, z)-separator $\{v\}$ of minimum size one and the (non-)strict (s, z)-cut $\{(\{s, v\}, i) \in E \mid i \leqslant t\}$ of minimum size t.

Since it is NP-hard to decide if there is a constant number of vertex-disjoint (non-) strict (s, z)-paths in a temporal graph, there is no hope that the problem whether there are k vertex-disjoint (non-)strict (s, z)-paths in a temporal graph is fixed-parameter tractable when parameterized by k. Hence, this is presumably not an efficient way to compute a lower bound on the minimum size of a (non-)strict (s, z)-separator.

Now, we discuss an upper bound for the minimum size of a (non-)strict (s, z)-separator. In the proof of the next lemma, Theorem 3.4, and Proposition 3.14 we will make use of the NP-complete Vertex Cover problem [Kar72].

Vertex Cover
Input: A graph $G = (V, E)$ and $k \in \mathbb{N}$.
Question: Is there a subset $V' \subseteq V$ of size at most k such that for all $\{v, w\} \in E$ it holds $\{v, w\} \cap V' \neq \emptyset$?

Let $(G = (V, E), k)$ be a Vertex Cover instance. We say that $V' \subseteq V$ is a *vertex cover* in G of size k if $|V'| = k$ and V' is a solution to $(G = (V, E), k)$.

Lemma 2.8. *Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$. The minimum size of a (non-)strict (s, z)-separator is at most the minimum size of a (non-)strict (s, z)-cut.*

Proof. Let $G = (V, E, \tau)$ be a temporal graph, $s, z \in V$, and $X \subseteq E$ be a (non-) strict (s, z)-cut. We know that the temporal graph $(V, E \setminus X, \tau)$ has no (non-)strict (s, z)-path. Now consider the underlying graph H of $G[X]$. Let S be a vertex cover of H. Note that H has at most one edge between two vertices and that H has at most $|X|$ edges in total. We can create a vertex cover of size at most $|X|$ by iterating over the edges of H. For each $\{v, w\} \in E(H)$, take v or w into the vertex cover and delete all incident edges from of v or w, respectively. Hence, $|S| \leqslant |X|$. The graph H becomes edgeless if we delete S from H. Therefore, G does not contain any time-edge which corresponds to an edge in X if we delete S from G. Thus, $G - S$ has no non-strict (s, z)-path. This completes the proof for non-strict (s, z)-paths. The same argument holds for strict (s, z)-paths. □

This gap between the minimum size of a (non-)strict (s, z)-separator and the minimum size of a (non-)strict (s, z)-cut can also be arbitrarily large. To see this, consider the temporal graph $G = (\{s, z, v\}, E, t + 1)$ in Figure 2.5, with time-edges $(\{s, v\}, i)$ and $(\{v, t\}, i+1)$ for all $i \in \{1, \ldots, t\}$. Clearly, the minimum (non-)strict (s, z)-separator is of size one but the minimum (non-)strict (s, z)-cut is of size t.

However, the minimum size of a (non-)strict (s, z)-cut and hence an upper bound on the minimum size of a (non-)strict (s, z)-separator can be computed in polynomial-time.

Lemma 2.9 (Berman [Ber96])**.** *Let $G = (V, E, t)$ be a temporal graph and $s, z \in V$. The maximum number of edge-disjoint (non-)strict (s, z)-paths is equal to the minimum size of a (non-)strict (s, z)-cut and can be computed in $\mathcal{O}(k \cdot |E|)$ time.*

This lemma was proven by Berman [Ber96] in terms of edge-scheduled networks which is a model essentially equivalent to temporal graphs [KKK02].

We close this section by presenting a polynomial-time algorithm to compute an upper-bound on the minimum size of a (non-)strict (s, z)-separator which is at most $k \cdot \tau$, where k is the minimum size of a (non-)strict (s, z)-separator and τ the maximum label.

Observation 2.10. *Let $G = (V, E, \tau)$ be a temporal graph, $s, z \in V$, and $k \in \mathbb{N}$. If G has a (non-)strict (s, z)-separator of size at most k, then a (non-)strict (s, z)-separator of size at most $k \cdot \tau$ can be computed in $\mathcal{O}(|V| \cdot |E|^2 + (|V| \cdot |E|)^2)$.*

Proof. Let $G = (V = \{s = v_1, v_2, \ldots, v_n = z\}, E, \tau)$ be a temporal graph, X be a (non-) strict (s, z)-separator of size at most k, and $H = (S, A)$ be the (non-)strict static expansion of (G, s, z). One can observe that the vertex set $X' := \{u_{j,i} \in S \mid v_i \in X \text{ and } j \in \{1, \ldots, \tau\}\}$ is of size at most $k \cdot \tau$ and a (s, z)-separator in H.

Furthermore, H is a directed graph and hence, we can computed the minimum (s, z)-separator by an algorithm by Orlin [Orl13] in $\mathcal{O}(|S| \cdot |A|) = \mathcal{O}((|V| \cdot |E|) \cdot (|E| + |V| \cdot |E|)) = \mathcal{O}(|V| \cdot |E|^2 + (|V| \cdot |E|)^2)$. □

Chapter 3

Natural Parameters

In this chapter, we study (Non-)Strict (s, z)-Separation with respect to parameters that are natural to consider. First, we show in Section 3.1 that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size k. In the introductory example of Chapter 1, this is the number of administrators of the manufacturing company and thus, can be very small in real-world scenarios.

Second, we study the computational complexity of (Non-)Strict (s, z)-Separation when parameterized by the maximum label τ. In the introductory example of Chapter 1, this is the time between Patch Tuesday and the weekend, because we can update the system z on the backbone of our manufacturing process only on weekends. We believe that this parameter can also be very small in many cases. For example, we could alter the introductory scenario from Chapter 1 such that it takes only one hour of preparation to update the system z on the backbone of our manufacturing process. One can observe that this parameter depends on the precision of the discretization of time.

We prove that Non-Strict (s, z)-Separation is NP-hard, even if $\tau = 2$, and that Strict (s, z)-Separation is NP-hard, even if $\tau = 5$. Afterwards, we present polynomial-time algorithms for Strict (s, z)-Separation when $\tau \leqslant 4$ and for Non-Strict (s, z)-Separation when $\tau = 1$.

Third, we consider (Non-)Strict (s, z)-Separation when parameterized by the solution size k and the maximum label τ and present a fixed-parameter algorithm for Strict (s, z)-Separation which relies on an upper bound on the length of a strict (s, z)-path. Hence, we can also derive a fixed-parameter algorithm for (Non-)Strict (s, z)-Separation, when parameterized by the number $|V|$ of vertices, as well as when parameterized by the solution size k, the maximum label τ, and the maximum number $|V_c|$ of vertices in a connected component over all layers.

In this work, we leave open the question whether Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ.

3.1 Solution Size

In this section, we show that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size k.

First, we will introduce a few problems which will be helpful to show hardness for some of our temporal graph separation problems.

Length-Bounded (s, z)-Separation

Input: A graph $G = (V, E)$, two distinct vertices s, z, and integers ℓ, k.
Question: Is there a subset $X \subseteq (V \setminus \{s, z\})$ of size at most k such that there is no (s, z)-path in $G - X$ of length at most ℓ?

Note that there is also a cut variant of this problem.

Length-Bounded (s, z)-Cut

Input: A graph $G = (V, E)$, two distinct vertices s, z, and integers ℓ, k.
Question: Is there a subset $Y \subseteq E$ of size at most k such that there is no (s, z)-path $G \setminus Y$ of length at most ℓ?

Length-Bounded (s, z)-Separation (Length-Bounded (s, z)-Cut) is studied by Baier et al. [Bai+10] under the name L-length-bounded node cut (L-length-bounded edge cut). Golovach and Thilikos [GT11] studied Length-Bounded (s, z)-Separation and Length-Bounded (s, z)-Cut under the name Bounded Vertex Undirected (s, t)-Cut and Bounded Edge Undirected (s, t)-Cut, respectively. Fluschnik et al. [Flu+16] studied at Length-Bounded (s, z)-Cut and conjectured that most of their results hold for Length-Bounded (s, z)-Separation. Since we call path elimination problems by edge deletion cuts and by vertex deletion separators, a new name for this problem was necessary. Both problems are essentially equivalent with respect to the solution size k and the maximum length of a path ℓ [GT11].

First, we are reduce from Length-Bounded (s, z)-Separation to Strict (s, z)-Separation with respect to the solution size k. Afterwards, we show a reduction from Strict (s, z)-Separation to Non-Strict (s, z)-Separation with respect to the solution size k and the maximum label τ.

Theorem 3.1. Strict (s, z)-Separation *is* W[1]*-hard, when parameterized by the solution size* k*.*

Proof. We reduce from Length-Bounded (s, z)-Separation. Golovach and Thilikos [GT11] showed that this problem is W[1]-hard with respect to k. Let $I := (G = (V, E), s, z, \ell, k)$ be a Length-Bounded (s, z)-Separation instance. We construct a Strict (s, z)-Separation instance $\hat{O} := (\hat{G} = (V, \hat{E}, \ell), s, z, k')$ where $\{v, w\} \in E$ if and only if $(\{v, w\}, i) \in \hat{E}$ for all $i \in \{1, \ldots, \ell\}$. Set the parameter k' to k. Since $\ell \leqslant |V|$, this can be computed in polynomial-time. One can observe that for every path in G of length at most ℓ there is at least one corresponding strict path in $\hat{G}$ and that for each strict path in $\hat{G}$ there is a path in G of length at most ℓ. Therefore, $X \subseteq (V \setminus \{s, z\})$ is a solution for I if and only if X is solution for $\hat{O}$. Note that $k' = k$, hence this is a parameterized reduction with respect to k. □

Now, we reduce from Strict (s, z)-Separation to Non-Strict (s, z)-Separation to show W[1]-hardness for Non-Strict (s, z)-Separation when parameterized by the solution size k.

Corollary 3.2. Non-Strict (s, z)-Separation *is* W[1]*-hard, when parameterized by the solution size* k*.*

Proof. The idea is to reduce from Strict (s,z)-Separation and replace each edge by two vertices of degree two to make sure that two incident time-edges cannot be used.

Let $I := (G = (V, E, \tau), s, z, k)$ be a Strict (s,z)-Separation instance. We construct a Non-Strict (s,z)-Separation instance $\hat{O} := (G' = (V', E', 2\tau), s, z, k)$ where $V' := V \cup \{e_1, e_2 \mid e \in E\}$ and for each $(\{v,w\}, t) \in E$ we introduce the following time-edges into E':

$$(\{v, e_1\}, 2t-1), (\{e_1, w\}, 2t), (\{w, e_2\}, 2t-1), (\{e_2, v\}, 2t).$$

The vertices in $\{e_1, e_2 \mid e \in E\}$ are called *edge-vertices*. Note that this can be done in $\mathcal{O}(|V|+2\cdot|E|+|E|) = \mathcal{O}(|V|+|E|)$ time, that $|V'| = |V|+2\cdot|E|$, and that $|E'| = 4\cdot|E|$. Let $v, w \in V$. Observe that for each strict (v,w)-path

$$P = (\{v, v_1\}, t_1), \ldots, (\{v_{i-1}, v_i\}, t_i), \ldots, (\{v_{\ell-1}, w\}, t_\ell)$$

of length ℓ in G there is a strict (v,w)-path

$$\begin{aligned}\hat{P} = {} & (\{v, e_1\}, 2t_1 - 1), (\{e_1, v_1\}, 2t_1), \ldots, \\ & (\{v_{i-1}, e_i\}, 2t_i - 1), (\{e_i, v_i\}, 2t_i), \ldots, \\ & (\{v_{\ell-1}, e_\ell\}, 2t_\ell - 1), (\{e_\ell, w\}, 2t_\ell)\end{aligned}$$

of length 2ℓ in G', where $e_1, \ldots, e_\ell$ are edge-vertices. Furthermore, there is no time-edge $(\{v,w\}, t) \in E'$, therefore each non-strict (v,w)-path P' in G' has to visit at least one edge-vertex. Each edge-vertex has exactly two incident time-edges where both have distinct labels. Hence we can conclude that P' must be a strict (v,w)-path. For each sequence $(\{v_i, e_i\}, 2t_i-1), (\{e_i, v_{i+1}, 2t_i+1)$ in P' there is a time-edge $(\{v_i, v_{i+1}\}, t_i) \in E$, where $v_i, v_{i+1} \in V$ and e_i is an edge-vertex. Thus, we can construct a strict (v,w)-path in G from P'. See Figure 3.1 for an example.

Now we are going to show that there is a strict (s,z)-separator in G if and only if there is a non-strict (s,z)-separator in G'.

$\Rightarrow$: Let $S \subseteq V$ be a strict (s,z)-separator of size at most k in G. We have already shown for each strict (s,z)-path G there is a strict (s,z)-path in G' and that there is no non-strict (s,z)-path in G' which is not a strict (s,z)-path. This implies that S is a non-strict (s,z)-separator of size at most k in G'.

$\Leftarrow$: Let $S \subseteq V$ be a non-strict (s,z)-separator of size at most k in G', $e_i \in S$ be an edge-vertex, and $v \in N(e_i)$. Each non-strict (s,z)-path which visits e_i also visits v because an edge-vertex has degree two. Therefore, $(S \setminus \{e_i\}) \cup \{v\}$ is also a non-strict (s,z)-separator in G'. As a preprocessing step we replace each edge-vertex in S which one of its neighbors to get a non-strict (s,z)-separator $S' \subseteq V$ of size at most k. For each strict (s,z)-path G there is strict (s,z)-path in G' and each non-strict (s,z)-separator is also a strict (s,z)-separator. Hence, S' is a strict (s,z)-separator in G, because $S' \subseteq V$. □

Theorem 3.1 and Corollary 3.2 imply that (Non-)Strict (s,z)-Separation is in the complexity class W[1] or higher up in the hierarchy of hardness classes for parameterized problems. On the other hand, we can observe that the (Non-)Strict (s,z)-Separation is in W[P]. The class W[P] is the class of parameterized problems where

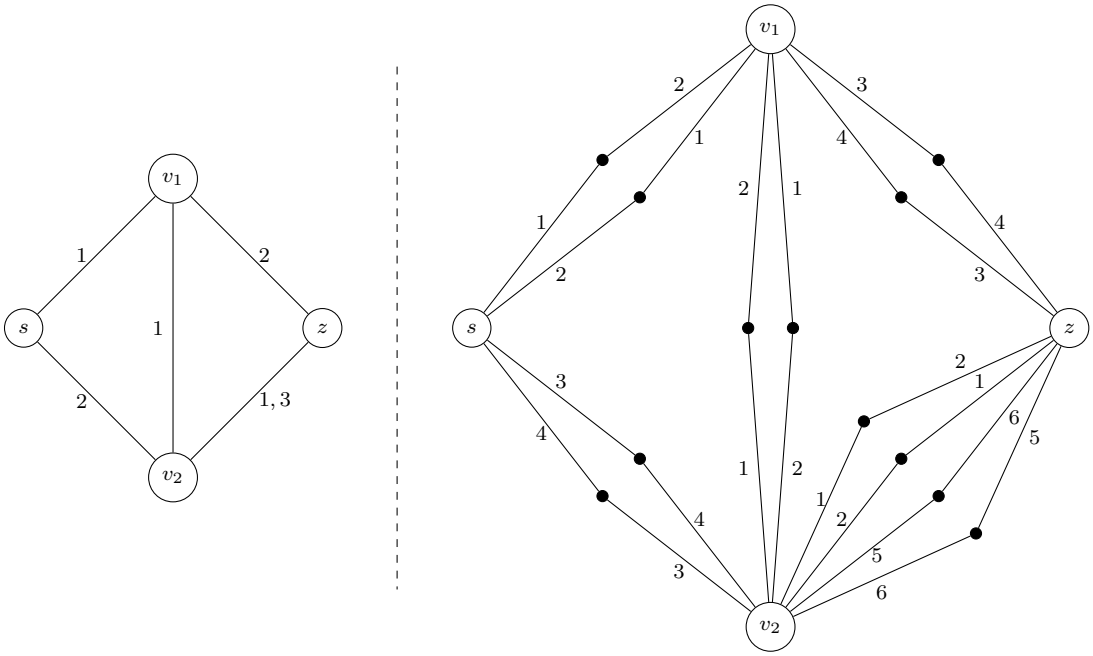


Figure 3.1: An example for the reduction in Corollary 3.2 of a temporal graph from a Strict (s, z)-Separation instance (left) to a temporal graph for a Non-Strict (s, z)-Separation instance (right). The edge-vertices are drawn by black dots. For each time-edge between vertices v, w in the Strict (s, z)-Separation instance with label t there is a strict (v, w)-path and a strict (w, v)-path with departure time $2t$ and arrival time $2t + 1$.

each instance (I, r) can be decided in $h(r) \cdot |I|^{\mathcal{O}(1)}$ time by a non-deterministic Turing-machine that makes at most $f(r) \cdot \log |I|$ non-deterministic choices, where h and f are computable functions [FG06, Theorem 25.2.1].

Observation 3.3. (Non-)Strict (s, z)-Separation *is in* W[P] *when parameterized by the solution size* k.

Proof. Let $I := (G = (V, E, \tau), s, z, k)$ be a (Non-)Strict (s, z)-Separation instance. Without loss of generality, we assume that each vertex $v \in V$ has a binary encoding and hence, can be addressed with $\log |I|$ bits. The non-deterministic Turing-machine guesses a set $S \subseteq V$ of at most k vertices. Note that these are $k \cdot \log |I|$ many non-deterministic choices. Observe that I is a no-instance if and only if we find a (non-)strict (s, z)-path in $G - S$. This can be checked in polynomial-time by the algorithm from Lemma 2.4. It follows that (Non-)Strict (s, z)-Separation is in W[P] when parameterized by the solution size k. □

3.2 Maximum Label

In this section we study the parameterized complexity of (Non-)Strict (s, z)-Separation when parameterized by the maximum label τ. First, we show that Strict

(s,z)-Separation is NP-hard, even if $\tau = 5$, and Non-Strict (s,z)-Separation is NP-hard, even if $\tau = 2$. This means that (Non-)Strict (s,z)-Separation is (probably) not in XP and hence, there is no fixed-parameter algorithm for (Non-)Strict (s,z)-Separation when parameterized by the maximum label τ.

Second, we discuss the open cases of Non-Strict (s,z)-Separation when the maximum label is $\tau = 1$ and of Strict (s,z)-Separation when the maximum label is $\tau = 1,2,3,4$ by showing polynomial-time algortihms for all of these cases.

Theorem 3.4. Strict (s,z)-Separation *is* NP*-hard, even if* $\tau = 5$*, and* Non-Strict (s,z)-Separation *is* NP*-hard, even if* $\tau = 2$*.*

Proof. Baier et al. [Bai+10] showed that Length-Bounded (s,z)-Separation is NP-hard, even if $\ell = 5$. Let $I := (G, s, z, \ell, k)$ be a Length-Bounded (s,z)-Separation instance. First, we use the reduction from Theorem 3.1 to compute an equivalent Strict (s,z)-Separation instance $O_1 := (G_1 = (V_1, E_1, \tau_1), s_1, z_1, k_1)$ in polynomial-time, where $\tau_1 = \ell$. This proves the first part of the theorem.

For the second part of the theorem, we are going to refine the gadget of Baier et al. [Bai+10] and reduce directly from Vertex Cover to Non-Strict (s,z)-Separation. Let $I := (G = (V,E), k)$ be a Vertex Cover instance and $n := |V|$. We construct a Non-Strict (s,z)-Separation instance $\hat{O} := (\hat{G} = (\hat{V}, \hat{E}, 2), s, z, k+n)$, where

$$\hat{V} := V \cup \{v_1, v_2 : v \in V\} \cup \{s, z\}$$

are the vertices and the time-edges are defined as

$$\hat{E} := \overbrace{\{(\{s, v_1\}, 1), (\{v_1, v\}, 1), (\{v, v_2\}, 2), (\{v_2, z\}, 2), (\{s, v\}, 2), (\{v, z\}, 1) : v \in V\}}^{\text{vertex-edges}} \cup \underbrace{\{(\{v_1, w_2\}, 1), (\{w_1, v_2\}, 1) : \{v, w\} \in E\}}_{\text{edge-edges}}.$$

For each vertex $v \in V$ there is a *vertex gadget* which consists of three vertices v_1, v, v_2 and six vertex-edges. In addition, for each edge $\{v, w\} \in E$ there is an *edge gadget* which consists of two edge-edges connecting v_1 with w_2 and v_2 with w_1. See Figure 3.2 for an example. The Non-Strict (s,z)-Separation instance $\hat{O}$ can be computed in polynomial-time, because $|\hat{V}| = 3 \cdot n + 2$ and $|\hat{E}| = 6 \cdot |\hat{V}| + 2 \cdot |E|$.

It remains to be shown that there is a vertex cover for G of size at most k if and only if there is a non-strict (s,z)-separator in $\hat{G}$ of size at most $n+k$.

$\Rightarrow$: Assume that $V' \subseteq V$ is a vertex cover of size $k' \leqslant k$ for G. Now, we build a non-strict (s,z)-separator $S := (V \setminus V') \cup \{v_1, v_2 \mid v \in V'\}$. There are $|V| - |V'| = n - k'$ vertices not in the vertex cover V' and for each of them there is exactly one vertex in S. For each vertex in the vertex cover V' there are two vertices in S. Hence, $|S| = n - k' + 2k' \leqslant n + k$. First, we consider just the vertex-gadget of a vertex $v \in V$. Note that there are two distinct non-strict (s,z)-separators in the vertex-gadget of v. One is $\{v\}$, and the other is $\{v_1, v_2\}$. Therefore, we can already observe that there is no non-strict (s,z)-path in $\hat{G} - S$ which does not contain an edge-edge. Second, let $e = \{v, w\} \in E$ and let P_e and P'_e be the non-strict (s,z)-paths which contain the edge-edges of edge-gadget of e such that $V(P_e) = \{s, v_1, w_2, z\}$ and $V(P'_e) = \{s, w_2, v_1, z\}$. Since V' is a vertex cover

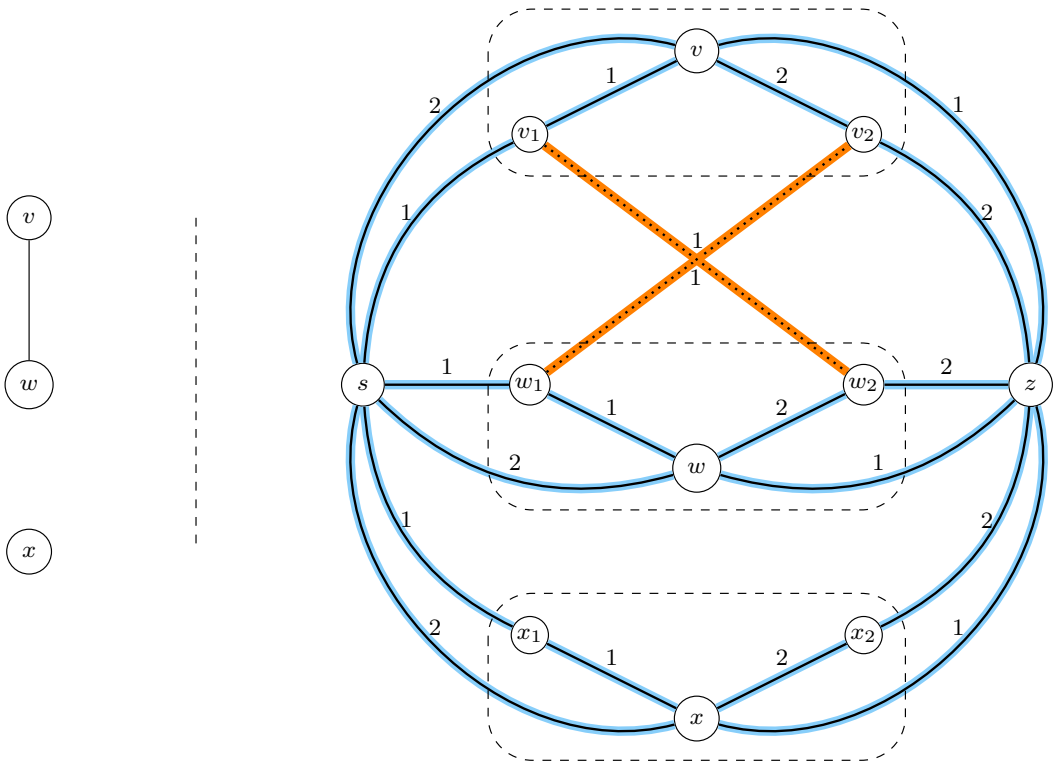


Figure 3.2: The Vertex Cover instance $(G, 1)$ (left) and the corresponding Non-Strict (s, z)-Separation instance from the reduction of Theorem 3.4 (right). The edge-edges are orange (dotted), the vertex-edges are blue (solid), and the vertex gadgets are in dashed boxes.

of G we know that at least one element of e is in V'. Without loss of generality, we assume $v \in V'$. Thus, $v_1, v_2 \in S$ and therefore neither P_e nor P'_e do exist in $\hat{G} - S$. This completes this direction, because there are no other non-strict (s, z)-path in $\hat{G}$.

$\Leftarrow$: Assume that there is a non-strict (s, z)-separator S in $\hat{G}$ of size $n + k' \leqslant n + k$ and let $v \in V$. Recall that there are two distinct non-strict (s, z)-separators in the vertex gadget of v, namely $\{v\}$ and $\{v_1, v_2\}$ and that all vertices in $\hat{V} \setminus \{s, z\}$ are from a vertex gadget. We start with a preprocessing to ensure that for vertex gadget only one of these two separators are in S. Let $S_v = S \cap \{v, v_1, v_2\}$. We iterate over S_v for each $v \in V$ and

Case 1: If $S_v = \{v\}$ or $S_v = \{v_1, v_2\}$ then we do nothing.

Case 2: If $S_v = \{v, v_1, v_2\}$ then we remove v from S and decrease k' by one. One can observe that all non-strict (s, z)-paths which are visiting v are still separated by v_1 or v_2.

Case 3: If $S_v = \{v, v_1\}$ then we remove v from S and add v_2. One can observe that S is still a non-strict (s, z)-separator of size k in $\hat{G}$.

Case 4: If $S_v = \{v, v_2\}$ then we remove v from S and add v_1. One can observe that S is still a non-strict (s, z)-separator of size k in $\hat{G}$.

That is a valid distinction of cases because v_1 or v_2 alone do not separate all non-strict (s, z)-paths in the vertex gadget in v. Now we construct a vertex cover V' for G by taking v into V' if both v_1 and v_2 are in S. The size of V' is $|S| - n = k' \leqslant k$, because there are n vertex gadgets in $\hat{G}$ and each vertex gadget containing either one or two nodes from S.
Assume towards a contradiction that V' is not a vertex cover of G. Therefore, there is an edge $\{v, w\} \in E$ where $v, w \notin V'$. Hence, $v_1, v_2, w_1, w_2 \notin S$ and $v, w \in S$. This contradicts that S is a non-strict (s, z)-separator in $\hat{G}$, because there is the non-strict (s, z)-path $P_{\{v,w\}} = ((\{s, v_1\}, 1), (\{v_1, w_2\}, 1), (\{w_2, z\}, 2))$. □

Theorem 3.4 does not settle the computational complexity of the Non-Strict (s, z)-Separation problem when $\tau = 1$ and of the Strict (s, z)-Separation problem when $\tau = 1, 2, 3, 4$.

A Non-Strict (s, z)-Separation instance where $\tau = 1$ has only one layer which implies that there is no temporal component left in the problem. Hence we can solve the problem by checking whether there is an (s, z)-separator of size k on the layer one.

Lemma 3.5 (Ford and Fulkerson [FF56])**.** *Let $G = (V, E)$ be a (directed) graph, $s, z \in V$ two distinct vertices, and $k \in \mathbb{N}$. It is decidable in $\mathcal{O}(k \cdot (|V| + |E|))$ time whether there is an (s, z)-separator of size at most k in G.*

Proof. Let $G = (V, E)$ be a (directed) graph, $s, z \in V$ two distinct vertices, and $k \in \mathbb{N}$. We construct a directed graph $D := (H, A_1 \cup A_2)$ such that

- $H := \{v_1, v_2 \mid v \in V\}$,
- $A_1 := \{(v_1, v_2) \mid v \in V\}$, and
- $A_2 := \{(x_2, y_1) \mid \{x, y\} \in E(G)\}$ (if G is a directed graph $A_2 := \{(x_2, y_1) \mid (x, y) \in E(G)\}$).

Now, we define a weight function

$$\omega : A_1 \cup A_2 \to \{1, \infty\}, (x, y) \mapsto \begin{cases} 1, \text{if } (x, y) \in A_1 \\ \infty, \text{otherwise} \end{cases}$$

and run k rounds of the algorithm by Ford and Fulkerson [FF56] on D with source s_2, sink z_1, and weight ω. One can observe that D has a (s_2, z_1)-cut C of weight $\sum_{e \in C} \omega(e) = k$ if and only if there is an (s, z)-separator of size k in G [MOR13]. The residual graph from Ford and Fulkerson [FF56] contains an (s_2, z_1)-path if and only if the maximum flow from s_2 to z_1 is at least $k + 1$. If the reader is not familiar with the definition of maximum flow, then we refer to Ford and Fulkerson [FF56]. By the Max-Flow-Min-Cut Theorem [FF56], the maximum flow from s_2 to z_1 is equal to the weight of the minimum (s_2, z_1)-cut.

The running time of each round of the algorithm by Ford and Fulkerson [FF56] can be upper-bounded by in $\mathcal{O}(|H| + |A_1| + |A_2|)$. We can find an (s_2, z_1)-path in the

residual graph by a breadth-first search which also runs in $\mathcal{O}(|H| + |A_1| + |A_2|)$ time. Hence, the algorithm runs in $\mathcal{O}(k \cdot (|V| + |E|))$ time, because $|H| = 2 \cdot |V|$, $|A_1| = |V|$, and $|A_2| = 2 \cdot |E|$. □

If the maximum label $\tau = 1$ in a temporal graph, then we can interpret the temporal graph as a standard graph. Hence, we have the following observation.

Observation 3.6. Non-Strict (s, z)-Separation *can be solved in* $\mathcal{O}(k \cdot (|V| + |E|))$ *time when the maximum label* $\tau = 1$.

Hence, Non-Strict (s, z)-Separation is polynomial-time solvable for $\tau = 1$ and becomes NP-hard even if $\tau = 2$.

The case of Strict (s, z)-Separation where $\tau = 1$ is trivial, because then all strict (s, z)-paths have length one, and thus, there must be a time-edge $(\{s, z\}, 1)$. We excluded this case by assumption for the whole thesis, because these are trivial no-instances. Nonetheless, we can test if there is such a time-edge by iterating over the time-edges set.

Observation 3.7. *The* Strict (s, z)-Separation *problem for* $\tau = 1$ *can be solved in* $\mathcal{O}(|E|)$ *time.*

The case of Strict (s, z)-Separation where $\tau = 2$ can be solved by the following data reduction rule.

Reduction Rule 3.1. *Let* $(G = (V, E, \tau), s, z, k)$ *be a* (Non-)Strict (s, z)-Separation *instance and let* P *be a (non-)strict* (s, z)*-path of length two where* $V(P) \setminus \{s, z\} = \{v\}$. *Then delete* v *and decrease* k *by one.*

It is quite easy to see that v is the one and only option to separate s from z on P. Hence v must be in every (non-)strict (s, z)-separator of G. A (non-)strict (s, z)-path can be found in $\mathcal{O}(|V| \cdot \tau + |E|)$ time by the breadth-first search from Lemma 2.4, but here we store for each node of the breadth-first search how many non-column edges we need to reach this node. Furthermore, we ensure that we do not explore paths which have more than two non-column edges. Consequently, we get only (non-)strict (s, z)-paths of length two.

It is rather obvious that we can decide the Strict (s, z)-Separation problem where $\tau = 2$ in $\mathcal{O}(k \cdot (|V| + |E|))$ time by executing Reduction Rule 3.1 at most k times and check afterwards if there is still a strict (s, z)-path in the temporal graph.

Observation 3.8. *The* Strict (s, z)-Separation *problem for* $\tau = 2$ *can be solved in* $\mathcal{O}(k \cdot (|V| + |E|))$ *time.*

To study the cases of Strict (s, z)-Separation where $\tau \in \{3, 4\}$, we introduce two more data reduction rules. Note that these data reduction rules are also useful for Non-Strict (s, z)-Separation and for an arbitrary maximum label τ.

Reduction Rule 3.2. *Let* $(G = (V, E, \tau), s, z, k)$ *be a* (Non-)Strict (s, z)-Separation *instance, and* $v \in V$. *If there is no (non-)strict* (s, z)*-path* P *with* $v \in V(P)$, *then delete* v.

To see the correctness of this rule, let G be a temporal graph, $G' = G - \{v\}$ be the temporal graph after the application of Reduction Rule 3.2, and let S be a (non-) strict (s, z)-separator in G'. Then, S is also a (non-)strict (s, z)-separator in G, because v is not visited by any (non-)strict (s, z)-path. Furthermore, if S' is a (non-)strict (s, z)-separator in G, then $S' \setminus \{v\}$ is a (non-)strict (s, z)-separator in G' because G' is a temporal subgraph of G.

Reduction Rule 3.3. *Let $(G = (V, E, \tau), s, z, k)$ be a* (Non-)Strict (s, z)-Separation *instance, and $e \in E$. If there is no (non-)strict (s, z)-path P which contains e, then delete e.*

To see the correctness of this rule, let G be a temporal graph, $G' = G \setminus e$ be the temporal graph after the application of Reduction Rule 3.3, and let S be a (non-)strict (s, z)-separator in G'. Then, S is also a (non-)strict (s, z)-separator in G, because there is no (non-)strict (s, z)-path which contains e. Furthermore, if S' is a (non-)strict (s, z)-separator in G, then S' is a (non-)strict (s, z)-separator in G' because G' is a temporal subgraph of G.

Lemma 3.9. *Let $(G = (V, E, \tau), s, z, k)$ be an instance of* (Non-)Strict (s, z)-Separation. *Reduction Rules 3.2 and 3.3 can be exhaustively applied in $\mathcal{O}(|V| \cdot \tau + |E|) \leqslant \mathcal{O}(|V| \cdot |E|)$ time.*

Proof. We provide an algorithm to apply Reduction Rules 3.2 and 3.3 simultaneously and exhaustively.

First, we show the algorithm for Strict (s, z)-Separation. Second, we explain how the algorithm must be altered to work for Non-Strict (s, z)-Separation. Let $I = (G = (V, E, \tau), s, z, k)$ be a Strict (s, z)-Separation input instance.

(i) Construct the strict static expansion $H = (S, A)$ of (G, s, z).

(ii) Perform a breadth-first search in H from s and mark all vertices in the search tree as *reachable*. Let $R(s) \subseteq S$ be the reachable vertices from s.

(iii) Construct $H' := (R(s), A')$, where $A' := \{(w, v) : (v, w) \in A \text{ and } v, w \in R(s)\}$. Observe that H' is the reachable part of H from s, where all directed arcs change their direction.

(iv) If $z \notin R(s)$, then our instance I is a yes-instance.

(v) Perform a breadth-first search from z in H' and mark all vertices in the search tree as reachable. Let $R(z) \subseteq R(s)$ be the reachable set of vertices from z. In the graph $H[R(s) \cap R(z)] = H[R(z)]$, all vertices are reachable from s and from each vertex the vertex z is reachable.

(vi) Output the temporal graph $G' := (V', E', \tau)$, where $V' := \{v_i \in V \mid \exists j : x_{j,i} \in R(z)\}$ and $E' := \{(\{v_j, v_{j'}\}, i) : (x_{(i-1),j}, x_{i,j'}) \in E(H[R(z)])\}$.

One can observe that G' is a temporal subgraph of G. Aside from the two breadth-first searches (ii),(v) and the construction of the strict static expansion (i), all subroutines are computable in linear time. By Lemma 2.3, (i) can be done in $\mathcal{O}(|V| \cdot \tau + |E|)$ time. The

running time estimations of the breadth-first searches in (ii) and (v) work identically to Lemma 2.4 and is therefore $\mathcal{O}(|V| \cdot \tau + |E|)$. Because of Lemma 2.2, we have $\mathcal{O}(|V| \cdot \tau + |E|) \leqslant \mathcal{O}(|V| \cdot |E|)$. Consequently, G' can be computed in $\mathcal{O}(|V| \cdot \tau + |E|) \leqslant \mathcal{O}(|V| \cdot |E|)$ time.

We are about to show that Reduction Rules 3.2 and 3.3 are not applicable in G'. Let $v_j \in V'$ and assume towards a contradiction that there is no strict (s, z)-path P in G' such that $v_j \in V(P)$. Since $v_j \in V$, there is an i such that there is an $(s, x_{i,j})$-path in H. Note that, because of the column-edges, $R(s)$ contains all $x_{i',j}$ where $i < i' \leqslant \tau$. Furthermore, there must be an $x_{i',j} \in R(z)$ where $i \leqslant i' \leqslant \tau$, also because of $v_j \in V'$ and hence there is an (s, z)-path P' in H such that $x_{i',j} \in V(P')$. This is a contradiction because then there is strict (s, z)-path in G as well as in G'—Reduction Rule 3.2 is not applicable.

Let $e = (\{v_j, v_{j'}\}, i) \in E'$ and assume towards a contradiction that there is no strict (s, z)-path P in G' which contains e. From the construction of G' we know that $(x_{(i-1),j}, x_{i,j'}) \in E(H[R(z)])$ or $(x_{(i-1),j'}, x_{i,j}) \in E(H[R(z)])$. Similarly as we have concluded before, there is an (s, z)-path in H which contains either $\{x_{(i-1),j}, x_{i,j'}\}$ or $\{x_{(i-1),j'}, x_{i,j}\}$.

This implies that there is a strict (s, z)-path in G as well as in G'. This is a contradiction. Reduction Rule 3.3 thus is not applicable.

It remains to be shown that each strict (s, z)-path P in G is also a strict (s, z)-path in G'. Since P is a strict (s, z)-path in G there is a corresponding (s, z)-path P' in H. Note that P' is a witness that all vertices in $V(P')$ are in $R(s)$ as well as in $R(z)$. It follows from the construction of G' that P is also a strict (s, z)-path in G'.

This algorithm works also for Non-Strict (s, z)-Separation when we construct the static expansion instead of the strict static expansion. □

Lovász, Neumann-Lara, and Plummer [LNLP78] showed that the minimum size of an (s, z)-separator for paths of length at most four in a graph is equal to the number of vertex-disjoint (s, z)-paths of length at most four in a graph. We are about to adjust their idea to strict paths on temporal graphs. The idea behind Lemma 3.11 is similar to the idea of the proof of Lovász, Neumann-Lara, and Plummer [LNLP78] and based on the following helper graph. However, since the connectivity by strict paths between vertices is not transitive, see Remark 2.1, we have to cover a lot more cases than Lovász, Neumann-Lara, and Plummer [LNLP78].

Definition 3.10. Let $G = (V, E, \tau = 4)$ be a temporal graph, $s, z \in V$ two distinct vertices, and Reduction Rules 3.1 to 3.3 are not applicable on G. The *directed path cover graph* from s to z is a directed graph $D = (V, \vec{E})$ such that $(v, w) \in \vec{E}$ if and only if

(i) $v, w \in V$,

(ii) $(\{v, w\}, t) \in E$, where t can be arbitrary,

(iii) $v \in V_{(i,j)}$ and $w \in V_{(i',j')}$ such that $i < i'$ or $v \in V_{(2,2)}$ and $w \in V_{(2,1)}$,

where a vertex $x \in V$ is in the set $V_{(i,j)}$ if the shortest strict (s, x)-path is of length i and the shortest strict (x, z)-path is of length j.

The directed path cover graph of a temporal graph with maximum time-edge label four has the following property.

Lemma 3.11. *Let $G = (V, E, \tau = 4)$ be a temporal graph, $s, z \in V$ two distinct vertices, Reduction Rules 3.1 to 3.3 are not applicable on G, and let $D = (V, \vec{E})$ be the directed path cover graph from s to z of G. Then $S \subseteq V \setminus \{s, w\}$ is a strict (s, z)-separator in G if and only if S is an (s, z)-separator on D.*

Proof. Let $G = (V, E, \tau = 4)$ be a temporal graph, $s, z \in V$ two distinct vertices, Reduction Rules 3.1 to 3.3 are not applicable on G, and let $D = (V, \vec{E})$ be the directed path cover graph from s to z of G.

We say that a strict (s, z)-path P of length n in G is *chordless* if $G[V(P)]$ does not contain a strict (s, z)-path P' of length $n - 1$. If such a P' exists, we call P' a *chord* of P. Obviously, for a set X of vertices, if $G - X$ does not contain any chordless strict (s, z)-path, then $G - X$ does not contain any strict (s, z)-path.

Now, we discuss the appearance of the directed path cover graph D. Assume towards a contradiction that there is a vetex $v \in V_{(1,1)}$. Due to Reduction Rule 3.1 there are no strict (s, z)-paths of length two. Thus, if there is a vertex $v \in V_{(1,1)}$, then there must be time-edges $(\{s, v\}, t_1)$ and $(\{v, z\}, t_2)$ such that $t_2 \leqslant t_1$. We distinguish two cases.

Case 1: Let $t_2 \leqslant t_1 \leqslant 2$. Since Reduction Rule 3.3 is not applicable, we know that there is a strict (s, z)-path P which contains $(\{v, z\}, t_2)$. The strict (s, z)-path P must be of length at least three, because otherwise Reduction Rule 3.1 would be applicable. Observe that the length of a strict (s, z)-path is a lower bound for the arrival time. Hence, the arrival time of P is at least three. This is a contradiction because $(\{v, z\}, t_2)$ is the last time-edge of P and therefore, $t_2 \leqslant 2$ is equal to the arrival time. Consequently, $v \notin V_{(1,1)}$.

Case 2: Let $3 \leqslant t_2 \leqslant t_1$. Since Reduction Rule 3.3 is not applicable, we know that there is a strict (s, z)-path P which contains $(\{s, v\}, t_1)$. The strict (s, z)-path P must be of length at least three, because otherwise Reduction Rule 3.1 would be applicable. Observe that $(\{s, v\}, t_1)$ is the first time-edge of P. Hence, the departure time of P is at least three. Since, P is of length at least three, the arrival time of P is at least five. This is a contradiction because the maximum label is four. Consequently, $v \notin V_{(1,1)}$.

Observe that the sets $V_{(1,3)}, V_{(2,2)}, V_{(3,1)}, V_{(1,2)}, V_{(2,1)}$ are a partition of V, because all strict (s, z)-paths are of length at most four and $V_{(1,1)} = \emptyset$. In Figure 3.3, we can see the appearance of the directed path cover graph D from s to z of G.

We claim that for each chordless strict (s, z)-path P in G, there is an (s, z)-path P_D such that $V(P) = V(P_D)$. Let P be a chordless strict (s, z)-path in G. Due to Reduction Rule 3.1, P is of length three or four.

Case 1: Let the length of P be three and $V(P) = \{s, v_1, v_2, z\}$ such that v_1 is visited before v_2. Since there is no strict (s, z)-path of length two, P forces v_1 into $V_{(1,2)}$ and $v_2 \in V_{(2,1)}$. It is easy to see that $v_1 \in V_{(1,2)}$, just by the existence of P. That $v_1 \in V_{(1,2)}$ is, just by the existence of P, easy to see. For $v_2 \in V_{(2,1)}$, we need that there is no strict (s, v_2)-path of length one.

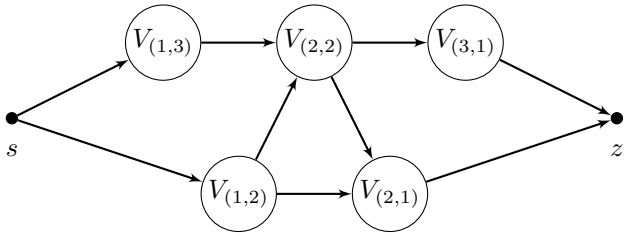


Figure 3.3: The directed path cover graph D from s to z of a temporal graph with maximum time-edge label $\tau = 4$. An arc from vertex set $V_{(i,j)}$ to vertex set $V_{(i',j')}$ denote that for two vertices $v \in V_{(i,j)}$ and $w \in V_{(i',j')}$ there can be an arc from v to w in D.

Assume towards a contradiction that there would be a strict (s, v_2)-path of length one or in other words there is a time-edge $(\{s, v_2\}, t) \in E$. All labels of the time-edge between v_2 and z are at least three thus t is at least three, otherwise there is a strict (s, z)-path of length two. There must be a strict (s, z)-path P' which uses the time-edge $(\{s, v_2\}, t)$, otherwise Reduction Rule 3.3 would be applicable. Furthermore, P' must be of length at least three, otherwise Reduction Rule 3.1 would be applicable. Thus, P' has departure time at most two. The time-edge $(\{s, v_w\}, t)$ is the first time-edge in P', because it is incident with s. Hence, $t < 2$. This contradicts the existence of a strict (s, v_2)-path of length one, and hence $v_2 \in V_{(2,1)}$.

From Definition 3.10, we know that there is an (s, z)-path P_D such that $V(P) = V(P_D)$.

Case 2: Let the length of P be four. Thus, it has the following appearance

$$P = (\{s, v_1\}, 1), (\{v_1, v_2\}, 2), (\{v_2, v_3\}, 3), (\{v_3, z\}, 4).$$

It follows immediately, that the shortest strict (s, v_1)-path is of length one. Furthermore, the shortest strict (s, v_2)-path is of length two, otherwise P would not be chordless.

Now, we claim that $v_3 \in V_{(2,1)} \cup V_{(3,1)}$. Assume towards a contradiction that $v_3 \notin V_{(2,1)} \cup V_{(3,1)}$. Observe that $v_3 \notin V_{(2,2)}$, because $(\{v_3, z\}, 4)$ is a shortest strict (v_3, z)-path of length one. Thus, there is a time-edge $(\{s, v_3\}, t) \in E$. This time-edge must be part of a strict (s, z)-path, otherwise Reduction Rule 3.3 is applicable. Since $v_2 \neq z$, we have $t \leqslant 3$. Note that $(\{s, v_3\}, t \leqslant 3), (\{v_3, z\}, 4)$ would be a strict (s, z)-path of length two. Hence, $(\{s, v_3\}, t) \notin E$ and $v_3 \in V_{(2,1)} \cup V_{(3,1)}$.

From Definition 3.10 we know that there is an (s, z)-path P_D such that $V(P) = V(P_D)$.

Let $S \subseteq (V \setminus \{s, z\})$ be an (s, z)-separator in D. Until now, we know that S is also a strict (s, z)-separator in G, because for each chordless strict (s, z)-path in G there is an (s, z)-path in D. Let $L \subseteq (V \setminus \{s, z\})$ be a strict (s, z)-separator in G. It remains to be shown that L is an (s, z)-separator in D.

Assume towards a contradiction that L is not an (s, z)-separator in D. Thus, there is an (s, z)-path P_D in $D - L$. The length of P_D is either three or four, see Figure 3.3.

Case 1: Let the length of P_D be three and $V(P_D) = \{s, v_1, v_2, z\}$ such that v_1 is visited before v_2. Thus, $v_1 \in V_{(1,2)}$ and $v_2 \in V_{(2,1)}$, see Figure 3.3. From Definition 3.10, we know that there are time-edges $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, z\}, t_3) \in E$. Note that each time-edge in G must be part of a strict (s, z)-path and that there are no strict (s, z)-paths of length two. Hence, $t_1 \in \{1, 2\}$, $t_2 \in \{2, 3\}$, and $t_3 \in \{3, 4\}$. Since $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, z\}, t_3)$ is not a strict (s, z)-path in G by our assumption, we have either $t_1 = t_2 = 2$ or $t_2 = t_3 = 3$.

Case 1.1: Let $t_1 = t_2 = 2$. The time-edge $(\{v_1, v_2\}, t_2 = 2)$ must be part of a strict (s, z)-path, and therefore we have either $(\{s, v_2\}, 1) \in E$ or $(\{s, v_1\}, 1) \in E$. As time-edge $(\{s, v_2\}, 1)$ cannot exist, $v_2 \in V_{(2,1)}$. Hence, $(\{s, v_1\}, 1) \in E$, $t_1 = 1 \neq 2$, $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, z\}, t_3)$ is a strict (s, z)-path in G, and $\{v_1, v_2\} \cap L \neq \emptyset$. This contradicts the existence of P_D in $D - L$.

Case 1.2: Let $t_2 = t_3 = 3$. There is a strict (s, z)-path which contains $(\{v_1, v_2\}, t_3 = 3)$. Since $t_3 = 3$, there is either a $(\{v_1, z\}, 4) \in E$ or $(\{v_2, z\}, 4) \in E$. The time-edge $(\{v_1, z\}, 4)$ cannot exist, because $v_1 \in V_{(1,2)}$. Hence $(\{v_2, z\}, 4) \in E$, $t_3 = 4 \neq 3$, $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, z\}, t_3)$ is a strict (s, z)-path in G, and $\{v_1, v_2\} \cap L \neq \emptyset$. This contradicts the existence of P_D in $D - L$.

Case 2: Let the length of P_D be four, and $V(P_D) = \{s, v_1, v_2, v_3, z\}$ such that v_1 is visited before v_2 and v_2 is visited before v_3. From Figure 3.3, one can observe that $v_1 \in V_{(1,3)} \cup V_{(1,2)}$ and $v_2 \in V_{(2,2)}$. From Definition 3.10, we know that there are time-edges $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, v_3\}, t_3), (\{v_3, z\}, t_4) \in E$. There is a strict (s, z)-path P which contains $(\{v_1, v_2\}, t_2)$, otherwise Reduction Rule 3.3 would be applicable. Since $v_2 \in V_{(2,2)}$, we know that P is of length four and the first two time-edges of P.

Now assume towards a contradiction that P visits v_2 before v_1. Hence, $(\{v_1, v_2\}, t_2)$ is the third time-edge in P. The first two time-edges of P are a strict (s, v_2)-path of length two and arrival time at least two. Thus, $t_2 \geqslant 3$. Observe that there is no strict (v_1, z)-path of length one and hence there is no time-edge $(\{v_1, z\}, 4)$, because $v_1 \in V_{(1,2)} \cup V_{(1,3)}$. This contradicts P visiting v_2 before v_1. Consequently, the strict (s, z)-path P visits v_1 before v_2 and $(\{v_1, v_2\}, t_2)$ is the second time-edge in P. Hence, $t_2 = 2$ and there is a time-edge $(\{s, v_1\}, 1)$ in P. This implies $t_1 = 1$.

The vertex $v_3 \in V_{(2,1)} \cup V_{(3,1)}$, see Figure 3.3. There is a strict (s, z)-path which contains $(\{v_2, v_3\}, t_3)$, otherwise Reduction Rule 3.3 would be applicable. Since $v_3 \in V_{(2,1)} \cup V_{(3,1)}$ and $v_2 \in V_{(2,2)}$, all strict (s, v_2)-paths and strict (s, v_3)-paths have length at least two and thus also arrival time at least two. Hence, $t_3 \geqslant 3$. Because $v_2 \neq z \neq v_3$ and there is a strict (s, z)-path which contains $(\{v_2, v_3\}, t_3)$, we have $t_3 < 4 \Rightarrow t_3 = 3$ and there is either a time-edge $(\{v_2, z\}, 4) \in E$ or $(\{v_3, z\}, 4) \in E$. The time-edge $(\{v_2, z\}, 4)$ cannot exist, otherwise there would be a strict (v_2, z)-path of length one, but $v_2 \in V_{(2,2)}$. This implies that $(\{v_3, z\}, 4) \in E$ and $t_4 = 4$. Hence, $(\{s, v_1\}, t_1), (\{v_1, v_2\}, t_2), (\{v_2, v_3\}, t_3), (\{v_3, z\}, t_4)$ forms a strict (s, z)-path and $\{v_1, v_2, v_3\} \cap L \neq \emptyset$. This contradicts the existence of P_D in $D - L$.

In summary, P_D cannot exist in $D - L$. Consequently, L is an (s, z)-separator in D. □

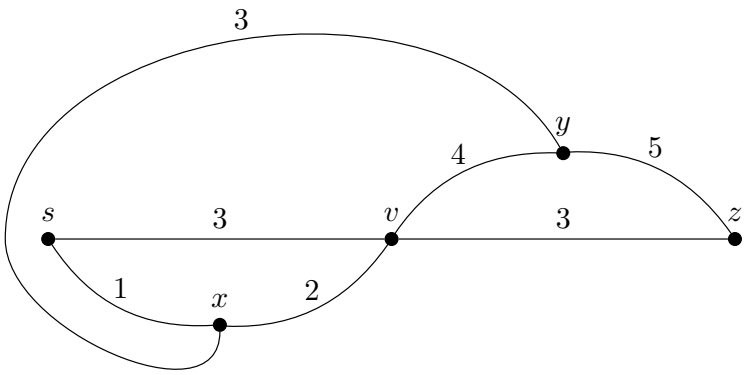


Figure 3.4: A temporal graph with maximum label $\tau = 5$ on which Lemma 3.11 does not hold. The vertex v would be in the vertex set $V_{(1,1)}$ which implies that v must be in an (s, z)-separator of the directed path cover graph. Hence, the strict (s, z)-separator $\{x, y\}$ in the temporal graph is not an (s, z)-separator in the directed path cover graph.

One can observe that Lemma 3.11 does not hold when the maximum label $\tau > 4$. We refer to Figure 3.4 for an example.

To construct the directed path cover graph D from s to z of G, we need to know the length of a shortest strict (s, v)-path and the length of a shortest strict (v, z)-path, for all $v \in V$. That is the single-source shortest path problem on temporal graphs.

Single-Source Shortest Strict Paths
Input: A temporal graph $G = (V, E, \tau)$, a vertex s
Task: Find the shortest strict (s, v)-path in G, for all $v \in V \setminus \{s\}$.

Lemma 3.12. Single-Source Shortest Strict Paths *can be solved in* $\mathcal{O}(|V| \cdot \tau + |E|)$ *time.*

Proof. In $\mathcal{O}(|V| \cdot \tau + |E|)$ time, we compute the strict static expansion $H = (S, A)$ of (G, s, z) and define a weight function

$$\omega : A \to \{0, 1\}, (x, y) \mapsto \begin{cases} 0, \text{if } (x, y) \text{ is a column-edge} \\ 1, \text{otherwise} \end{cases}$$

Observe that H with ω is a weighted directed acyclic graph and that the weight of an (s, z)-path in H with ω is equal to the length of the corresponding strict (s, z)-path in G. Hence, we can use an algorithm, which makes use of the topological order of S on H, to compute for all $v \in S$ a shortest (s, v)-path in H in $\mathcal{O}(|S| + |A|)$ time [Cor+09, Theorem 24.5]. For each $w_i \in V \setminus \{s, z\}$, there are τ vertices $w_{1,i}, \ldots, w_{\tau,i}$ in H. Note that shortest $(s, w_{j,i})$-path over all $j \in \{1, \ldots, \tau\}$ in H corresponds to a shortest strict (s, w_i)-path in G. Hence, for each vertex $w_i \in V \setminus \{s, z\}$, we can find the shortest strict (s, w_i)-path in G in $\mathcal{O}(\tau)$ time, if we already know all shortest paths from s in H.

The overall running time is

$$\mathcal{O}(|S| + |A| + |V| \cdot \tau) = \mathcal{O}((|V| \cdot \tau) + (|V| \cdot \tau + |E|) + |V| \cdot \tau) = \mathcal{O}(|V| \cdot \tau + |E|).$$

□

Note that Wu et al. [Wu+14] developed, in a slightly different model, algorithms which are similar to ours of Lemma 3.12. In their model of a temporal graph, a time-edge is directed and must have different incoming and outgoing labels. Therefore a path in their model is akin with a strict path in our model.

We can use the directed path cover graph and Lemma 3.11 to solve Strict (s, z)-Separation in polynomial-time, when the maximum time-edge label τ is at most four.

Theorem 3.13. Strict (s, z)-Separation *for maximum label* $\tau = 4$ *can be solved in* $\mathcal{O}(|E| \cdot |V|^2)$ *time.*

Proof. Let $I := (G = (V, E, \tau = 4), s, z, k)$ be an instance of Strict (s, z)-Separation. At first we exhaustively apply Reduction Rules 3.1 to 3.3 in $\mathcal{O}(k \cdot (|V| \cdot \tau + |E|))$ time on G, see Lemma 3.9.

To construct the directed path cover graph D from s to z of G, we compute the length of a shortest strict (s, v)-path and the length of a shortest strict (v, z)-path by Lemma 3.12, for all $v \in V$. Let $v \in V$. To compute the length of the shortest strict (v, z)-path in G, we construct the strict static expansion H_v of (G, v, z) and execute the algorithm of Lemma 3.12 again. Thus, we can compute the directed path cover graph D from s to z of G in $\mathcal{O}(|V| \cdot (|V| \cdot \tau + |E|))$ time. By Lemma 3.5, we can check whether D has an (s, z)-separator of size k in $\mathcal{O}(k \cdot (|V| + |E|)$ time. Note that $\tau = 4$. Hence, the overall running time is $\mathcal{O}(|E| \cdot |V|^2)$. □

Hence, Non-Strict (s, z)-Separation is polynomial-time solvable for $\tau \leqslant 4$ and becomes NP-hard even if $\tau = 5$. Nevertheless, the running time can be improved if the maximum time-edge label is $\tau = 3$.

Proposition 3.14. Strict (s, z)-Separation *with maximum label* $\tau = 3$ *can be solved in* $\mathcal{O}(k \cdot |V| \cdot |E|)$ *time.*

Proof. Let $(G = (V, E, \tau = 3), s, z, k)$ be an instance of Strict (s, z)-Separation. First of all, there is no strict (s, z)-path of length at least four in G. As a preprocessing, Reduction Rules 3.1 to 3.3 are exhaustively applied. Without loss of generality assume that there is still a strict (s, z)-path in G, otherwise $(G = (V, E, \tau = 3), s, z, k)$ is a trivial yes-instance. Because of Reduction Rule 3.1 there is no strict (s, z)-path of length two in G, and because of Reduction Rule 3.2 each vertex in G is visited by a strict (s, z)-path. One can observe that each strict (s, z)-path P has the following appearance $(\{s, v_1\}, 1), (\{v_1, v_2\}, 2), (\{v_2, z\}, 3)$. First, there is a time-edge with label one. Second, there is a time-edge with label two. Third, there is a time-edge with label three.

A *bipartite graph (or triangle-free graph)* has a partition of the vertex set into two sets such that the induced graph of each partition set is edgeless.

We claim that the layer two of G, namely G_2, is a bipartite graph. Assume towards a contradiction that G_2 is not a bipartite graph. Thus, there are vertices $v, w, x \in V$ such that $G_2[\{v, w, x\}]$ is a triangle. Since Reduction Rule 3.3 is not applicable, we have

- either a strict (s, z)-path $P_{e_1} = (\{s, v\}, 1), (\{v, w\}, 2), (\{w, z\}, 3)$, or $P'_{e_1} = (\{s, w\}, 1), (\{w, v\}, 2), (\{v, z\}, 3)$,
- either a strict (s, z)-path $P_{e_2} = (\{s, v\}, 1), (\{v, x\}, 2), (\{x, z\}, 3)$, or $P'_{e_2} = (\{s, x\}, 1), (\{x, v\}, 2), (\{v, z\}, 3)$, and
- either a strict (s, z)-path $P_{e_3} = (\{s, w\}, 1), (\{w, x\}, 2), (\{x, z\}, 3)$, or $P'_{e_3} = (\{s, x\}, 1), (\{x, w\}, 2), (\{w, z\}, 3)$.

Note that the P_{e_1} and P'_{e_1} cannot exist simultaneously, because otherwise there would be a strict (s, z)-path of length two which visits v or w. The same holds for P_{e_2} and P'_{e_2} and also for P_{e_3} and P'_{e_3}. Hence, there are eight cases in which $P_1 \in \{P_{e_1}, P'_{e_1}\}$, $P_2 \in \{P_{e_2}, P'_{e_2}\}$, and $P_3 \in \{P_{e_3}, P'_{e_3}\}$ exist in G. One can observe that in all these cases there is a vertex $y \in \{v, w, x\}$ such that there are time-edges $(\{s, y\}, 1), (\{y, z\}, 3) \in E$. This is a contradiction because $(\{s, y\}, 1), (\{y, z\}, 3)$ would be a strict (s, z)-path of length two in G. Thus, G_2 is a bipartite graph.

We claim that each strict (s, z)-separator S is a vertex cover of G_2. Since there are no strict (s, z)-paths of length two, we know that each vertex $v \in S$ is incident with an edge of layer two. Assume towards a contradiction that S is not a vertex cover of G_2. Thus, there is an edge $\{v, w\}$ in $G_2 - S$. There must be strict (s, z)-path $(\{s, v\}, 1), (\{v, w\}, 2), (\{w, z\}, 3)$ or $(\{s, w\}, 1), (\{v, w\}, 2), (\{v, z\}, 3)$, otherwise Reduction Rule 3.3 would be applicable. This contradicts S being a strict (s, z)-separator in G.

Finally, we can compute a vertex cover of a bipartite graph with König's theorem (see [Die16]) and an algorithm by Hopcroft and Karp [HK73] in $\mathcal{O}(|E| \cdot \sqrt{|V|})$ time, because Reduction Rules 3.1 to 3.3 are exhaustively applied in $\mathcal{O}(k \cdot (|V| \cdot \tau + |E|)) \leqslant \mathcal{O}(k \cdot (|V| \cdot |E|))$ time (cf. Lemma 3.9 and Observation 3.8). This gives an overall time complexity of $\mathcal{O}(k \cdot |V| \cdot |E|)$. □

3.3 Solution Size and Maximum Label

In this section, we consider the combined parameter of the solution size k and the maximum label τ. We show that Strict (s, z)-Separation is fixed-parameter tractable when parameterized by $k + \tau$. The idea behind this algorithm also leads to fixed-parameter algorithms for (Non-)Strict (s, z)-Separation when parameterized by the number $|V|$ of vertices.

In the end, we discuss whether Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ.

Theorem 3.15. Strict (s, z)-Separation *can be solved in* $\mathcal{O}(\tau^{k+3} \cdot |V| + |E|)$ *time, where* k *is the solution size,* τ *is the maximum label,* $|V|$ *is the number of vertices, and* $|E|$ *is the number of time-edges.*

Proof. We present a depth-first search algorithm (see Algorithm 1) to show fixed-parameter tractability. Let $I := (G = (V, E, \tau), s, z, k)$ be a Strict (s, z)-Separation instance. The basic idea of this algorithm is simple: at least one vertex of each strict (s, z)-path must be in the strict (s, z)-separator. Thus, we compute an arbitrary strict (s, z)-path (Line 4) and branch over all visited vertices of that strict (s, z)-path (Line 9) until we

Algorithm 1: The algorithm of Theorem 3.15 to compute a strict (s, z)-separator of a temporal graph.

Data: Temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and integer $k \in \mathbb{N}$.
Result: A strict (s, z)-separator of G.

```
1  getSeparator(∅, k);
2  output no;
3  function getSeparator(S,k)
4      compute strict (s, z)-path P in G − S;
5      if there is no strict (s, z)-path in G − S then
6          output yes;
7          exit;
8      else if k > 0 then
9          for v ∈ V(P) \ {s, z} do
10             getSeparator(S ∪ {v}, k − 1);
11         end
12     end
```

cannot find a strict (s, z)-path in $G - S$ or until we already picked k vertices to be in strict (s, z)-separator. Hence, if the algorithm outputs yes, then S is a strict (s, z)-separator.

We claim that if there is a strict (s, z)-separator in G, then Algorithm 1 outputs yes. Let S' be a strict (s, z)-separator in G. Let P_1 be the first strict (s, z)-path computed by Line 4. Then there is a vertex v such that $v \in (V(P_1) \setminus \{s, z\}) \cap S'$, otherwise S' is not a strict (s, z)-separator in G. Thus, at some point the algorithm chooses the vertex v in the for-loop in Line 9. Observe that the function call *getSeparator*$(\{v\}, k-1)$ in G is equivalent to a call *getSeparator*$(\emptyset, k')$ on $G - \{v\}$, where $k' = k - 1$. Hence, from here one can argue inductively that the algorithm choose the correct vertex in each recursive call. This implies the correctness of the algorithm.

From Lemma 2.4, we know that we can compute Line 4 in $\mathcal{O}(|V| \cdot \tau + |E|)$ time. Now, we upper-bound the size of the search tree in which each node is a call of the *getSeparator()* function. We can upper-bound the maximum depth of the search tree by $k + 1$ as in each recursive call we decrease k by one, until $k = 0$. Furthermore, the length of a strict (s, z)-path P is at most τ. Hence, there are at most $\tau - 1$ vertices in $V(P)$ which implies that each node in the search tree has at most $\tau - 1$ child nodes. Thus we can upper-bound the running time of Algorithm 1 by

$$\mathcal{O}((\tau - 1)^{k+2} \cdot |V| \cdot \tau + |E|) = \mathcal{O}(\tau^{k+3} \cdot |V| + |E|).$$

□

Note that in an implementation of Algorithm 1 it would not make sense to pick a degree-two vertex $v \in V(P)$ in Line 4 if there is a vertex $w \in V(P)$ of degree at least three, because both v and w cover the strict (s, z)-path P, but in a sharp contrast to v, the vertex w might also cover other strict (s, z)-paths. However, there are temporal graphs in which each vertex has degree at least three and hence, this does not give us a better upper bound on the running time.

One can observe that the length of a strict (s, z)-path is essential to bound the running time of this algorithm. We can upper-bound the length of a non-strict (s, z)-path by $|V|-1$. In standard graph theory we can upper-bound the size of the graph $G = (V, E)$, and therefore the input size for many graph problems, by $\mathcal{O}(|V| + |V|^2)$. This is not the case for temporal graphs, because the maximum size of a temporal graph also depends on the number of layers. Hence, the number $|V|$ of vertices in a temporal graph is an interesting parameter for which the algorithmic idea of Theorem 3.15 is also applicable.

Corollary 3.16. (Non-)Strict (s, z)-Separation *can be solved in* $\mathcal{O}((|V| - 2)^{k+2} \cdot |V| \cdot \tau + |E|)$ *time, where* k *is the solution size,* $|V|$ *is the number of vertices, and* $|E|$ *is the number of time-edges.*

Since $k \leqslant |V| - 2$, (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the number $|V|$ of vertices.

Corollary 3.17. (Non-)Strict (s, z)-Separation *can be solved in* $\mathcal{O}((|V| - 2)^{|V|} \cdot |V| \cdot \tau + |E|)$ *time, where* $|V|$ *is the number of vertices,* τ *the maximum label, and* $|E|$ *is the number of time-edges.*

Corollary 3.17 has the consequence that there are (presumably) no computable functions f, g and no set of reduction rules which can be applied in $g(\tau) \cdot |V| \cdot |E|$ time on a temporal graph $G = (V, E, \tau)$ such that it holds for the resulting equivalent temporal graph $G' = (V', E', \tau')$ that $|V'| \leqslant f(\tau)$, because this would be a polynomial-time algorithm for (Non-)Strict (s, z)-Separation, where the maximum label τ is constant. This would imply $\mathsf{P} = \mathsf{NP}$, because of Theorem 3.4.

The *maximum number* $|V_c|$ *of vertices in a connected component over all layer* of a temporal graph $G = (V, E, \tau)$ is defined as $|V_c| := \max_{i \in \{1,\dots,\tau\}} |V_i|$, where V_i is the maximum connected component of layer G_i, for all $i \in \{1, \dots, \tau\}$. Hence, we can upper-bound the length of a non-strict (s, z)-path by $\tau \cdot |V_c|$, because a non-strict (s, z)-path can visit at most $|V_c|$ many vertices in one layer and there are at most τ layers in G.

Corollary 3.18. (Non-)Strict (s, z)-Separation *can be solved in* $\mathcal{O}((\tau \cdot |V_c|)^{k+3} \cdot |V| + |E|)$ *time, where* k *is the solution size,* $|V|$ *is the number of vertices,* $|E|$ *is the number of time-edges, and* $|V_c|$ *is the maximum number of vertices in a connected component over all layers.*

It remains open whether Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ.

On the one hand, we are not able to present a W[1]-hardness reduction with respect to the solution size k for Non-Strict (s, z)-Separation where the maximum label τ is constant or also bounded by parameter of the W[1]-hardness reduction. Also, we are not able to present an NP-hardness reduction for Non-Strict (s, z)-Separation for constant k and τ.

On the other hand, it seems that none of the approaches of *important separators* [Mar06], *shadow removal* [MR14], *treewidth reduction* [MOR13], *LP-branching and CSP-branching* [Cyg+13; IWY16], *randomized contractions* [Chi+16], and *homogeneous path systems* [Gui11] are applicable to Non-Strict (s, z)-Separation. The major difficulty

is that connectivity is not transitive (see Remark 2.1) and that a non-strict (s, z)-separator can look very different from an (s, z)-separator (see Remark 2.2).

In the Section 4.1 we take a long journey through the discussion of the treewidth of a temporal graph. Originally, the intention for this was to lift the treewidth reduction technique to temporal graphs. This will not be achieved in this work and thus, can be seen as further research opportunity.

Chapter 4

Temporal Graph Classes

In this chapter, we study (Non-)Strict (s, z)-Separation on several (structurally) restricted temporal graphs.

In Section 4.1 we extend the concept of treewidth, see Definition 4.1, to temporal graphs. One way to do that is to consider the *underlying treewidth*, that is, the treewidth of the underlying graph of a temporal graph. In Section 4.1.1, we devise a fixed-parameter algorithm for (Non-)Strict (s, z)-Separation when parameterized by the underlying treewidth and the maximum label τ. Hence, (Non-)Strict (s, z)-Separation is in FPT on classes of temporal graphs of constant bounded underlying treewidth when parameterized by the maximum label. Furthermore, we discuss whether (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the underlying treewidth.

Another way to extend the concept of treewidth to temporal graphs is to consider the *layer treewidth*. That is, the maximum treewidth over all layers of a temporal graph. In Section 4.1.2, we use a technique by Mans and Mathieson [MM14] to show fixed-parameter tractability of Non-Strict (s, z)-Separation when parameterized by the solution size, the maximum label, and the layer treewidth. Hence, Non-Strict (s, z)-Separation is in FPT on classes of temporal graphs of constant layer treewidth when parameterized by the solution size and the maximum label. Afterwards, we discuss whether one of the parameters of the latter algorithm can be dropped.

In Section 4.3, we show that, unless NP $\subseteq$ coNP/poly, (Non-)Strict (s, z)-Separation does not admit a polynomial kernel when parameterized by any subset of the parameters

- solution size,
- maximum label,
- underlying treewidth,
- layer treewidth, and
- maximum degree of the temporal graph.

4.1 Treewidth of Temporal Graphs

One of the tools from the standard repertoire for designing fixed-parameter algorithms for graph problems is the tree-decomposition [Cyg+15; DF13; Die16; FG06; Nie06]. A

tree-decomposition is a mapping of a graph into a related tree-like structure. For many graph problems this tree-like structure can be used to formulate a bottom-up dynamic program that starts at the leaves and ends at the root of the tree-decomposition [Cyg+15; DF13; FG06; Nie06].

Definition 4.1. A *tree-decomposition* of a graph G is a pair $\mathcal{T} := (T, (B_i)_{i \in V(T)})$ consisting of a tree T and a family $(B_i)_{i \in V(T)}$ of *bags* $B_i \subseteq V(G)$, such that

(i) for all vertices $v \in V(G)$ the set $B^{-1}(v) := \{i \in V(T) \mid v \in B_i\}$ is non-empty and induces a subtree of T and

(ii) for every edge $e \in E(G)$ there is $i \in V(T)$ with $e \subseteq B_i$.

The *width* of $\mathcal{T}$ is $\max\{|B_i| - 1 \mid i \in V(T)\}$. The *treewidth* $\mathrm{tw}(G)$ of G is defined as the minimal width over all tree-decompositions of G.

For algorithmic purposes, it is important to compute tree-decompositions efficiently. In general this cannot be done in polynomial time with respect to the input size.

Theorem 4.2 (Arnborg, Corneil, and Proskurowski [ACP87])**.** *The problem to decide, given a graph G and $k \in \mathbb{N}$, whether $\mathrm{tw}(G) \leqslant k$ is* NP*-complete.*

However, there are fixed-parameter algorithms computing an optimal tree-decompositions in linear time for constant treewidth.

Theorem 4.3 (Bodlaender [Bod96])**.** *There is an algorithm which, on input G, computes a tree-decomposition of G of width $\mathrm{tw}(G)$ in $2^{\mathcal{O}(\mathrm{tw}(G)^3)} \cdot |V(G)|$ time.*

Hence, for a graph G of treewidth $\mathrm{tw}(G)$, we can first compute the tree-decomposition $\mathcal{T}$ of G in $f(\mathrm{tw}(G)) \cdot |V(G)|$ time and then solve the given problem on $\mathcal{T}$ in $g(\mathrm{tw}(G)) \cdot |V(G)|^{\mathcal{O}(1)}$ time, where f and g are computable functions. This would show fixed-parameter tractability of the given problem when parameterized by the treewidth.

We can derive two straightforward parameters from Definition 4.1 for temporal graphs: the *underlying treewidth* which is the treewidth of the underlying graph of a temporal graph, and the *layer treewidth* which is the maximum treewidth over all layers of a temporal graph. Note that the underlying treewidth is at least the layer treewidth, because each layer is a subgraph of the underlying graph. On the other hand, one can construct a temporal graph such that the underlying graph has arbitrarily large treewidth while each layer has constant treewidth. In Figure 4.1, we show a temporal graph with two layers which both have treewidth two, but the underlying graph contains a large grid and hence the underlying treewidth depends on the size of the grid. To see that both layers have treewidth two, let $G_i - \{z\}$ one of the layers without z. One can observe that $G_i - z$ is a tree and has therefore treewidth one. Now, we add z to every bag of a tree-decomposition of $G_i - \{z\}$ and obtain a tree-decomposition of width two for G_i.

In Section 4.1.1, we focus on the underlying treewidth and show that (Non-)Strict (s, z)-Separation in fixed-parameter tractable when parameterized by the underlying treewidth and the maximum label.

In Section 4.1.2, we focus on the layer treewidth and use a technique of Mans and Mathieson [MM14] to show fixed-parameter tractability of Non-Strict (s, z)-Separation, when parameterized by the solution size, the maximum label, and the layer treewidth. Afterwards, we show that we cannot drop the parameter solution size or maximum label from that algorithm.

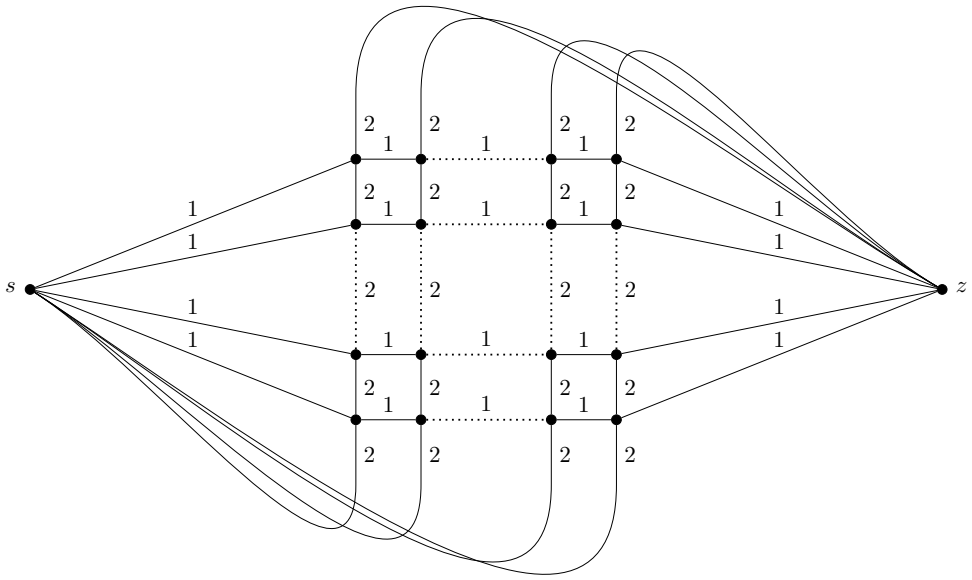


Figure 4.1: A temporal graph in which each layer has treewidth two but the underlying graph contains an arbitrarily large grid, and hence has an arbitrarily large treewidth.

4.1.1 Underlying Treewidth

In this section, we show that (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the underlying treewidth and the maximum label. Subsequently, we discuss, but leave open, whether (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized only by the underlying treewidth.

A tree-decomposition with a particularly simple structure is given by the following.

Definition 4.4. A tree-decomposition $\mathcal{T} := (T, (B_i)_{i \in V(T)})$ of a graph G is a *nice tree-decomposition* if the following conditions are satisfied:

(i) T is rooted.

(ii) Every node of the tree T has at most two children nodes.

(iii) If $i \in V(T)$ has two children nodes $k, j \in V(T)$ in T, then $B_i = B_k = B_j$. In this case i is called a *join node*.

(iv) If a node $i \in V(T)$ has one child node j, then one of the following conditions must hold:

(a) $B_i = B_j \cup \{v\}$. In this case i is called an *introduce node* of v.

(b) $B_i = B_j \setminus \{v\}$. In this case i is called a *forget node* of v.

(v) If node $i \in V(T)$ is a leaf in T, then $|B_i| = 1$. In this case i is called a *leaf node*.

Furthermore, for the node $i \in V(T)$, the tree T_i is the subtree of T rooted at i, that means T_i contains i itself and $j \in V(T)$ if every path from j to the root of T visits i. The set $B(T_i) := \bigcup_{j \in V(T_i)} B_j$ is the union of all bags of T_i.

It is not hard to transform a given tree-decomposition into a nice tree-decomposition. To be exact, the following result holds by Theorem 4.3 and Lemma 13.1.13 of Kloks [Klo94].

Lemma 4.5 (Kloks [Klo94])**.** *There is an algorithm which computes a nice tree-decomposition of a graph G of width* $\mathrm{tw}(G)$ *with* $\mathcal{O}(\mathrm{tw}(G) \cdot |V(G)|)$ *nodes in* $2^{\mathcal{O}(tw(G)^3)} \cdot |V(G)|$ *time.*

We formulate a dynamic program on the nice tree-decomposition of the underlying graph to solve (Non-)Strict (s, z)-Separation.

Theorem 4.6. (Non-)Strict (s, z)-Separation *is fixed-parameter tractable when parameterized by the underlying treewidth and the maximum label τ.*

The proof of Theorem 4.6 relies on a dynamic program on a nice tree-decomposition for the Non-Strict (s, z)-Separation. The dynamic program can also be applied to Strict (s, z)-Separation by the reduction from Corollary 3.2. First, we describe the dynamic program and give the proof of Theorem 4.6, subsequently.

We denote a sequence of sets $A_1, \ldots, A_n$ with $A_{[1:n]}$.

We are going to color V with $\tau + 2$ colors $\langle A_{[1:\tau]}, S, Z\rangle$. A vertex $v \in V$ of color $Y \in \{A_{[1:\tau]}, S, Z\}$ is denoted by $v \in Y$ and therefore each color is a set of vertices. The meaning of colors is that if $v \in S$, then v is in the non-strict (s, z)-separator, if $v \in Z$, then v is not reachable from s in $G - S$, and if $v \in A_i$, then v cannot be reached before time point i from s.

We say $\langle A_{[1:\tau]}, S, Z\rangle$ is a *coloring* of $X \subseteq V(G)$ if $X = A_1 \uplus \cdots \uplus A_\tau \uplus S \uplus Z$. A coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of vertex set $X \subseteq V(G)$ is *valid* if

(i) $s \in A_1$,

(ii) $z \in Z$,

(iii) for all $a \in A_i$, $a' \in A_j$, and $b \in Z$

- there is no non-strict (a, b)-path with departure time at least i in $G[X] - S$, and
- there is no non-strict (a, a')-path with departure time at least i and arrival time at most $j - 1$ in $G[X] - S$.

We call a coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of $X \subseteq Y \subseteq V(G)$ *extendible* to Y if there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of Y such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$.

Lemma 4.7. *Let $G = (V, E, \tau)$ be a temporal graph, and $s, z \in V$. There is a valid coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of V such that $|S| = k$ if and only if there is a non-strict (s, z)-separator S' of size k in G*

Proof. $\Rightarrow$: Assume that $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of V such that $|S| = k$. The vertex s has the color A_1 and the vertex z has the color Z. We know that there is no non-strict (s, z)-path in $G[V] - S = G - S$, otherwise (iii) of the definition of a valid coloring does not hold. Hence, the vertex set S is a non-strict (s, z)-separator of size k in G.

$\Leftarrow$: Assume that S' is a non-strict (s, z)-separator of size k in G. We construct a valid coloring as follows. All vertices in S' are of color S. Let $A \subseteq V(G)$ all from s reachable vertices in $G - S$. The vertices in the set $V(G) \setminus (A \cup S')$ are of color Z. Note that $z \in Z$. We set $s \in A_1$. Let $v \in A$ and $t \in \{1, \ldots, \tau\}$ be the earliest time point in which v can be reached from s. We set $v \in A_t$. As consequence there cannot be a $w \in A_{t'}$, such that there is a non-strict (w, v)-path with departure time at least t' and arrival time at most $t - 1$, because otherwise there would be a non-strict (s, v)-path with arrival time at most $t - 1$. This would imply that t is not the earliest time point in which v can be reached from s. Finally, we can observe that there are no $a \in A_i$ and $b \in Z$ such that there is a non-strict (a, b)-path with departure time at least i, because a can be reached at time point i from s and all vertices of color Z are not reachable in $G - S$. Hence, $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring for V. □

Let $(G = (V, E, \tau)$ be a temporal graph, $s, z \in V$, $G_\downarrow$ be the underlying graph of G, and $\mathcal{T} = (T, (B_i)_{i \in V(T)})$ be a nice tree-decomposition of $G_\downarrow$ of width $\mathrm{tw}(G_\downarrow)$. We add s, z to every bag of $\mathcal{T}$. Thus, $\mathcal{T}$ is of width at most $\mathrm{tw}(G_\downarrow) + 2$.

In the following, we give a dynamic program on $\mathcal{T}$. For each node x in T we compute a table D_x which stores for each coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x the minimum size of S' over all valid colorings $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_x)$ such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$.

$$D_x[A_{[1:\tau]}, S, Z] := \begin{cases} \min |S'|, & \text{there is a valid coloring } \langle A'_{[1:\tau]}, S', Z'\rangle \\ & \text{of } B(T_x) \text{ where } S \subseteq S', Z \subseteq Z', \text{ and } A_i \subseteq A'_i \\ & \text{for all } i \in \{1, \ldots, \tau\}. \\ \infty, & \text{otherwise} \end{cases} \tag{4.1}$$

Let $r \in V(T)$ be the root of T. If $D_r[A_{[1:\tau]}, S, Z] = k' < \infty$, then the coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_r is extendible to $B(T_r) = V(G)$ and there is a non-strict (s, z)-separator of size k' in G. Hence, the input instance I is a yes-instance if and only if $k' \leqslant k$.

The dynamic program first computes the tables for all leaf nodes of T and then in a "bottom-up" manner, all tables of nodes of which all child nodes are already computed. The computation of D_x, $x \in V(T)$, depends on the type of x, that is, whether x is a leaf, introduce, forget, or join node.

Leaf node. Let $x \in V(T)$ be a leaf node of $\mathcal{T}$. Thus, $B_x = \{s, v, z\}$. We test each coloring of B_x and set $D_x[A_{[1:\tau]}, S, Z] = \infty$ if $s \notin A_1$ or $z \notin Z$, because the coloring cannot be valid. Assume $s \in A_1$ or $z \in Z$. We distinguish three cases.

Case 1: If $v \in S$, then this is clearly a valid coloring. We set $D_x[A_{[1:\tau]}, S, Z] = 1$.

Case 2: If $v \in Z$, then we set

$$D_x[A_{[1:\tau]}, S, Z] = \begin{cases} \infty, & \text{if there is a } (\{s, v\}, t) \in E(G[B_x]), \\ 0, & \text{otherwise.} \end{cases}$$

Case 3: If $v \in A_i$, then we set

$$D_x[A_{[1:\tau]}, S, Z] = \begin{cases} \infty, & \text{if there is a } (\{s, v\}, t) \in E(G[B_x]) \text{ with } t < i, \\ \infty, & \text{if there is a } (\{v, z\}, t) \in E(G[B_x]) \text{ with } i \leqslant t, \\ 0, & \text{otherwise,} \end{cases}$$

where $i \in \{1, \ldots, \tau\}$.

Lemma 4.8. *Let G be a temporal graph and $\mathcal{T}$ be a tree-decomposition of G as described above, $x \in V(T)$ be a leaf node, and $\langle A_{[1:\tau]}, S, Z\rangle$ be a coloring of B_x. Then the following holds:*

(i) $D_x[A_{[1:\tau]}, S, Z] < \infty$ if and only if $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x.

(ii) If $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x, then $D_x[A_{[1:\tau]}, S, Z] = |S|$.

(iii) The table entry $D_x[A_{[1:\tau]}, S, Z]$ can be computed in $\mathcal{O}(|E|)$ time .

Proof. We start with (i).

$\Leftarrow$: Let $D_x[A_{[1:\tau]}, S, Z] = \infty$ and assume towards a contradiction that $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x. There are five cases in which $D_x[A_{[1:\tau]}, S, Z]$ is set to ∞. Either $s \notin A_1$, $z \notin Z$, $v \in Z$ and there is a time-edge $(\{s, v\}, t) \in E(G[B(T_x)])$, or $v \in A_i$ and there is a time-edge $(\{s, v\}, t) \in E(G[B(T_x)])$ with $t < i$ or there is a time-edge $(\{v, z\}, t) \in E(G[B(T_x)])$ with $i \leqslant t$, where $i \in \{1, \ldots, \tau\}$. All these cases contradict $\langle A_{[1:\tau]}, S, Z\rangle$ being a valid coloring of B_x.

$\Rightarrow$: Let $D_x[A_{[1:\tau]}, S, Z] < \infty$. Note that s must be of color A_1 and z must be of color Z. Observe that $D_x[A_{[1:\tau]}, S, Z] = 0$ or $D_x[A_{[1:\tau]}, S, Z] = 1$. Assume $D_x[A_{[1:\tau]}, S, Z] = 1$. Thus, $v \in S$. This implies that $G[B(T_x)] - S$ is time-edgeless and therefore $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x. Now assume $D_x[A_{[1:\tau]}, S, Z] = 0$. If $v \in Z$, then there is no time-edge from s to v which means $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x. If $v \in A_i$, then there is no time-edge $(\{s, v\}, t)$ such that $t < i$ and there is no time-edge from $(\{z, v\}, t)$ such that $i \leqslant t$. In both cases $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x.

Clearly, if $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x, then $D_x[A_{[1:\tau]}, S, Z] = |S|$ because we set $D_x[A_{[1:\tau]}, S, Z]$ only to one if $v \in S$. Furthermore, regardless of the coloring we can check by iterating over the time-edge set if $\langle A_{[1:\tau]}, S, Z\rangle$ is a valid coloring of B_x. Thus, (ii) and (iii) hold as well. □

Introduce node. Let $x \in V(T)$ be an introduce node of $\mathcal{T}$, $x' \in V(T)$ denote its child node, and $B_x \setminus B_{x'} = \{v\}$. We distinguish three cases.

Case 1: If $v \in S$, then we set $D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_{[1:\tau]}, S \setminus \{v\}, Z] + 1$.

Case 2: If $v \in Z$, then we set

$$D_x[A_{[1:\tau]}, S, Z] = \begin{cases} D_{x'}[A_{[1:\tau]}, S, Z \setminus \{v\}], & \text{if for all } w \in V \text{ with } (\{w,v\},t) \in E(G[B(T_x)]) \text{ holds, } w \in A_i \Rightarrow t < i, \\ \infty, & \text{otherwise.} \end{cases}$$

Case 3: If $v \in A_i$, then we set

$$D_x[A_{[1:\tau]}, S, Z] = \begin{cases} D_{x'}[A_{[1:i-1]}, A_i \setminus \{v\}, A_{[i+1:\tau]}, S, Z], & \text{if for all } (\{v,w\},t) \in E(G[B(T_x)]) \text{ holds: } t \geqslant i \Rightarrow w \in \bigcup_{j=1}^{t} A_j \cup S \text{ and } t < i \Rightarrow w \in \bigcup_{j=t+1}^{\tau} A_j \cup S \cup Z, \\ \infty, & \text{otherwise,} \end{cases}$$

where $i \in \{1, \ldots, \tau\}$.

First, we show the correctness of Case 1.

Lemma 4.9. *Let G and $\mathcal{T}$ be as described above, $x \in V(T)$ be an introduce node of v, $x' \in V(T)$ be the child node of x, $\langle A_{[1:\tau]}, S, Z\rangle$ be a coloring of B_x and $v \in S$. Then the following holds:*

(i) Coloring $\langle A_{[1:\tau]}, S \setminus \{v\}, Z\rangle$ of $B_{x'}$ is extendible to $B(T_{x'})$ if and only if coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is extendible to $B(T_x)$.

(ii) The value of $D_x[A_{[1:\tau]}, S, Z]$ corresponds to Equation (4.1) and can be computed in $\mathcal{O}(1)$ time.

Proof. $\Rightarrow$: Let $\langle A_{[1:\tau]}, S \setminus \{v\}, Z\rangle$ be a coloring of $B_{x'}$ which is extendible to $B(T_{x'})$. Then there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_{x'})$ such that $S \setminus \{v\} \subset S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$, where S' is a non-strict (s,z)-separator in $G[B(T_{x'})]$ of size $D_{x'}[A_{[1:\tau]}, S \setminus \{v\}, Z]$. Note that $v \notin S'$, because $v \notin B(T_{x'})$, because x is the introduce node for v (see (i) of Definition 4.1). Since $B(T_x) \setminus B(T_{x'}) = \{v\}$, we know that $G[B(T_{x'})] - S'$ is the same temporal graph as $G[B(T_x)] - (S' \cup \{v\})$. Hence, the coloring $\langle A_{[1:\tau]}, S, Z\rangle$ is extendible to $B(T_x)$ and $|S' \cup \{v\}| = |S'| + 1$ implies that the table entry $D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_{[1:\tau]}, S \setminus \{v\}, Z] + 1$.

$\Leftarrow$: Let $\langle A_{[1:\tau]}, S \setminus \{v\}, Z\rangle$ not be extendible to $B(T_{x'})$ then $\langle A_{[1:\tau]}, S, Z\rangle$ is not extendible to $B(T_x)$ because $G[B(T_{x'})]$ is a temporal subgraph of $G[B(T_x)]$, where $v \notin V(G[B(T_{x'})])$. Hence, $D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_{[1:\tau]}, S \setminus \{v\}, Z] + 1 = \infty + 1 = \infty$.

Note that $D_x[A_{[1:\tau]}, S, Z]$ can be computed in $\mathcal{O}(1)$ time because we just have to look up the value of $D_{x'}[A_{[1:\tau]}, S, Z]$. □

Second, we show the correctness of Case 2.

Lemma 4.10. *Let G and $\mathcal{T}$ be as described above, $x \in V(T)$ be an introduce node of v, $x' \in V(T)$ be the child node of x, $\langle A_{[1:\tau]}, S, Z\rangle$ be a coloring of B_x and $v \in Z$. Then the following holds:*

(i) Coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is extendible to $B(T_x)$ if and only if coloring $\langle A_{[1:\tau]}, S, Z \setminus \{v\}\rangle$ of $B_{x'}$ is extendible to $B(T_{x'})$ and for all $(\{w, v\}, t) \in E(G[B(T_x)])$ it holds that if $w \in A_i$ then $t < i$.

(ii) The value of $D_x[A_{[1:\tau]}, S, Z]$ corresponds to Equation (4.1) and can be computed in $\mathcal{O}(|E|)$ time.

Proof. $\Rightarrow$: Let coloring $\langle A_{[1:\tau]}, S, Z\rangle$ be extendible to $B(T_x)$. Then, there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_x)$ such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$. Since $B(T_{x'}) = B(T_x) \setminus \{v\}$ and $(Z \setminus \{v\}) \subseteq Z \subseteq Z'$, the coloring $\langle A_{[1:\tau]}, S, Z \setminus \{v\}\rangle$ of $B_{x'}$ is extendible to $B(T_{x'})$. Furthermore, $v \in Z$ implies that if there is a time-edge $(\{w, v\}, t) \in E(G[B(T_x)])$ then $t < i$.

$\Leftarrow$: First, if coloring $\langle A_{[1:\tau]}, S, Z \setminus \{v\}\rangle$ of $B_{x'}$ is not extendible to $B(T_{x'})$ then coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x cannot be extendible to $B(T_x)$ because $G[B(T_{x'})]$ is a temporal subgraph of $G[B(T_x)]$. Hence, $D_x[A_{[1:\tau]}, S, Z] = \infty$.

Let $\langle A_{[1:\tau]}, S, Z \setminus \{v\}\rangle$ be a coloring of $B_{x'}$ which is extendible to $B(T_{x'})$ and for all $(\{w, v\}, t) \in E(G[B(T_x)]$ it holds that $w \in A_i$ implies $t < i$. Assume towards a contradiction that coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is not extendible to $B(T_x)$. Since $\langle A_{[1:\tau]}, S, Z \setminus \{v\}\rangle$ is extendible to $B(T_{x'})$ we know that there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_{x'})$ such that $S \subseteq S'$, $Z \setminus \{v\} \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$. Therefore, $s \in A'_1$, $z \in Z'$, and for all $a \in A'_i$ and $a' \in A'_j$ there is no non-strict (a, a')-path with departure time at least i and arrival time at most $j-1$ in $G[B(T_{x'})] - S$, for all $i, j \in \{1, \ldots, \tau\}$. Thus, there must be an $a \in A'_i$ and $b \in Z'$ such that there is a non-strict (a, b)-path P in $G[B(T_x)] - S$ with departure time at least i, for some $i \in \{1, \ldots, \tau\}$. Since $B(T_x) \setminus B(T_{x'}) = \{v\}$, the vertex v must be the first vertex of color Z which is visited by P. Furthermore, there is a time-edge $(\{w, v\}, t) \in E(G[B(T_x)])$ such that $w \in A_i$, where $i \leqslant t$. This contradicts i being larger then t. Hence, $\langle A_{[1:\tau]}, S, Z\rangle$ is extendible to $B(T_x)$. Because $v \in Z$, we have $D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_{[1:\tau]}, S, Z \setminus \{v\}]$.

Note that $D_x[A_{[1:\tau]}, S, Z]$ can be computed in $\mathcal{O}(|E|)$ time, because we can iterate once over the time-edges set E and decide if for all $(\{w, v\}, t) \in E(G[B(T_x)]$ it holds that $w \in A_i$ implies $t < i$. □

Third, we show the correctness of Case 3.

Lemma 4.11. *Let G and $\mathcal{T}$ be as described above, $x \in V(T)$ be an introduce node of v, $x' \in V(T)$ be the child node of x, $\langle A_{[1:\tau]}, S, Z\rangle$ be a coloring of B_x and $v \in A_i$, where $i \in \{1, \ldots, \tau\}$. Then the following holds:*

(i) Coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is extendible to $B(T_x)$ if and only if coloring $\langle A_{[1:i-1]}, A_i \setminus \{v\}, A_{[i+1:\tau]}, S, Z\rangle$ of $B_{x'}$ is extendible to $B(T_{x'})$ and for each $(\{v, w\}, t) \in E(G[B(T_x)])$ it holds that

- *if* $t \geqslant i$ *then* $w \in \bigcup_{j=1}^{t} A_j \cup S$, *and*
- *if* $t < i$ *then* $w \in \bigcup_{j=t+1}^{\tau} A_j \cup S \cup Z$.

(ii) The value of $D_x[A_{[1:\tau]}, S, Z]$ *corresponds to Equation (4.1) and can be computed in* $\mathcal{O}(|E|)$ *time.*

Proof. $\Rightarrow$: Let coloring $\langle A_{[1:\tau]}, S, Z\rangle$ be extendible to $B(T_x)$. Then, there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_x)$ such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_j \subseteq A'_j$, for all $j \in \{1, \ldots, \tau\}$. Since $B(T_{x'}) = B(T_x) \setminus \{v\}$ and $(A_i \setminus \{v\}) \subseteq A_i \subseteq A'_i$, the coloring $\langle A_1, \ldots, A_i \setminus \{v\}, \ldots, A_\tau, S, Z\rangle$ of $B_{x'}$ is extendible to $B(T_{x'})$.

Let $(\{v, w\}, t) \in E(G[B(T_x)])$. We to distinguish into two cases. First, let $t \geqslant i$. Note that $w \in B_x$ because x is the introduce node for v. Since $\langle A'_{[1:\tau]}, S', Z'\rangle$ is a valid coloring of $B(T_x)$, $w \notin Z$ because there cannot be a non-strict (v, w)-path with departure time t in $G[B(T_x)] - S'$. Assume towards a contradiction that $w \in A_j$, where $j \in \{t+1, \ldots, \tau\}$. Then the time-edge $(\{v, w\}, t)$ is a non-strict (v, w)-path with departure time at least i and arrival time at most $j - 1$. Since, $\langle A'_{[1:\tau]}, S', Z'\rangle$ is a valid coloring of $B(T_x)$, such a non-strict (v, w)-path does not exist. This is a contradiction. Hence, $w \in \bigcup_{j=1}^{t} A_j \cup S$.

Second, let $t < i$. Again, $\langle A'_{[1:\tau]}, S', Z'\rangle$ is a valid coloring of $B(T_x)$ and therefore $w \notin \bigcup_{j=1}^{t} A_j$ because otherwise there would be a non-strict (w, v)-path in $G[B(T_x)] - S'$ with departure time at least t and arrival time $t < i$ and this would contradicts $\langle A'_{[1:\tau]}, S', Z'\rangle$ being a valid coloring. Hence $w \in \bigcup_{j=t+1}^{\tau} A_j \cup S \cup Z$.

$\Leftarrow$: First, if coloring $\langle A_1, \ldots, A_i \setminus \{v\}, \ldots, A_\tau, S, Z\rangle$ of $B_{x'}$ is not extendible to $B(T_{x'})$ then coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x cannot be extendible to $B(T_x)$ because $G[B(T_{x'})]$ is a temporal subgraph of $G[B(T_x)]$. Hence, $D_x[A_{[1:\tau]}, S, Z] = \infty$.

Let coloring $\langle A_1, \ldots, A_i \setminus \{v\}, \ldots, A_\tau, S, Z\rangle$ of $B_{x'}$ be extendible to $B(T_{x'})$ and for each $(\{v, w\}, t) \in E(G[B(T_x)])$ it holds that: if $t \geqslant i$ then $w \in \bigcup_{j=1}^{t} A_j \cup S$ and if $t < i$ then $w \in \bigcup_{j=t+1}^{\tau} A_j \cup S \cup Z$. Assume towards a contradiction that coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is not extendible to $B(T_x)$. The coloring $\langle A_1, \ldots, A_i \setminus \{v\}, \ldots, A_\tau, S, Z\rangle$ is extendible to $B(T_{x'})$ and therefore we have a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_{x'})$ such that $S \subseteq S'$, $Z \subseteq Z'$, $A_i \setminus \{v\} \subseteq A'_i$, and $A_j \subseteq A'_j$, for all $j \in \{1, \ldots, \tau\} \setminus \{i\}$. But $\langle A'_1, \ldots, A'_i \cup \{v\}, \ldots, S', Z'\rangle$ is not a valid coloring for $B(T_x)$. We know $s \in A'_1$ and $z \in Z'$. Thus there must be either an $a \in A'_j$ and $a' \in A'_\ell$ such that there is a non-strict (a, a')-path P_1 with departure time at least j and arrival time at most $\ell - 1$ or an $b \in Z$ such that there is a non-strict (a, b)-path P_2 with departure time at least j, for some $j, \ell \in \{1, \ldots, \tau\}$. We will show that P_1 and P_2 do not exist, and hence coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is extendible to $B(T_x)$. One can observe that this is a contradiction.

Suppose towards a contradiction that P_1 exists. Since coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_{x'})$ is valid, we know that P_1 visits v. Thus, there are time-edges $(\{w_1, v\}, t_1), (\{v, w_2\}, t_2) \in E(G[B(T_x)])$ in P_2 such that w_1 is visited before v and v is visited before w_2, where $w_1 \in A'_{u_1}$, $w_2 \in A'_{u_2}$. Refer to Figure 4.2 for an illustration.

We know $u_1 \leqslant t_1$ because there is no non-strict (a, w_1)-path with departure time at least j and arrival time at most $u_1 - 1$. We know $t_1 \geqslant i$ because otherwise $w_1 \in \bigcup_{j'=t_1+1}^{\tau} A_{j'}$, but $u_1 \leqslant t_1$. We know $i \leqslant t_2$ because P_1 is a non-strict (a, a')-path which implies $t_1 \leqslant t_2$. We know that $u_2 \leqslant t_2$ because $i \leqslant t_2$ and therefore the vertex $w_2 \in$

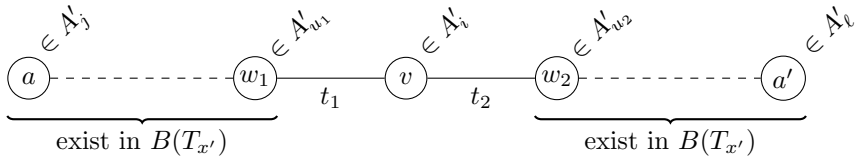


Figure 4.2: The non-strict (a, a')-path P_1 from the proof of Lemma 4.11. One can observe that $u_1, i \leqslant t_1 \leqslant t_2$ and $u_2 \leqslant t_2$.

$\bigcup_{j'=1}^{t_2} A_{j'}$ and $w_2 \in A_{u_2}$. This contradicts the existence of P_1 because there is a non-strict (w_2, a')-path with departure time at least $u_2 \leqslant t_2$ and arrival time $\ell - 1$. Otherwise, coloring $\langle A'_{[1:\tau]}, S', Z' \rangle$ of $B(T_{x'})$ is not valid. Hence, P_1 does not exist.

Now suppose towards a contradiction that P_2 exists. The vertex $v \in A_i$ is the last vertex visited by P_2 which is not colored by Z, otherwise we would be able to find a subsequence of P_2 similar to P_1. Thus, there are time-edges $(\{w_1, v\}, t_1), (\{v, b\}, t_2) \in E(G[B(T_x)])$ which are in P_2 such that w_1 is visited before v and v is visited before b, where $w_1 \in A'_{u_1}$. We conclude equivalent to the case of P_1 that $u_1 \leqslant t_1$, $i \leqslant t_1$, $i \leqslant t_2$. Therefore, we have $b \in \bigcup_{j=1}^{t} A_j \cup S$, because of the time-edge $(\{v, b\}, t_2)$ and $i \leqslant t_2$. This contradicts the existence of P_2, because $b \notin Z$.

Clearly, $D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_1, \ldots, A_i \setminus \{v\}, \ldots, A_\tau, S, Z]$ because $v \notin S$.

Note that $D_x[A_{[1:\tau]}, S, Z]$ can be computed in $\mathcal{O}(|E|)$ time, because we can iterate once over the time-edge set E and decide if for all $(\{w, v\}, t) \in E(G[B(T_x)]$ it holds that if $t \geqslant i$ then $w \in \bigcup_{j=1}^{t} A_j \cup S$ and if $t < i$ then $w \in \bigcup_{j=t+1}^{\tau} A_j \cup S \cup Z$. □

Forget node. Let $x \in V(T)$ be a forget node of $\mathcal{T}$, $x' \in V(T)$ its child, and $B_{x'} \setminus B_x = \{v\}$. We set

$$D_x[A_{[1:\tau]}, S, Z] = \min \begin{Bmatrix} \min_{i \in \{1,\ldots,\tau\}} D_{x'}[A_{[1:i-1]}, A_i \cup \{v\}, A_{[i+1,\tau]}, S, Z], \\ D_{x'}[A_{[1:\tau]}, S \cup \{v\}, Z], \\ D_{x'}[A_{[1:\tau]}, S, Z \cup \{v\}] \end{Bmatrix}.$$

Lemma 4.12. *Let G and $\mathcal{T}$ be as described above, $x \in V(T)$ be a forget node of v, $x' \in V(T)$ be the child node of x, and $\langle A_{[1:\tau]}, S, Z \rangle$ be a coloring of B_x. Then the following holds:*

(i) Coloring $\langle A_{[1:\tau]}, S, Z \rangle$ of B_x is extendible to $B(T_x)$ if and only if there is a coloring $\langle A'_{[1:\tau]}, S', Z' \rangle$ of $B_{x'}$ which is extendible to $B(T_{x'})$ such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \ldots, \tau\}$.

(ii) The value of $D_x[A_{[1:\tau]}, S, Z]$ corresponds to Equation (4.1) and can be computed in $\mathcal{O}(|E|)$ time.

Proof. $\Rightarrow$: Let coloring $\langle A_{[1:\tau]}, S, Z \rangle$ of B_x be extendible to $B(T_x)$. Then there is a valid coloring $\langle A''_{[1:\tau]}, S'', Z'' \rangle$ of $B(T_x)$ such that $S \subseteq S''$, $Z \subseteq Z''$, and $A_i \subseteq A''_i$, for all $i \in \{1, \ldots, \tau\}$. Since x' is a child of x and $B_x \subseteq B_{x'}$, we know that $B(T_x) = B(T_{x'})$ and therefore there is a coloring $\langle A'_{[1:\tau]}, S', Z' \rangle$ of $B_{x'}$ which is extendible to $B(T_{x'})$,

where $S' \subseteq S''$, $Z' \subseteq Z''$, and $A'_i \subseteq A''_i$, for all $i \in \{1, \dots, \tau\}$. It follows from $B_x \subseteq B_{x'}$, that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \dots, \tau\}$.

$\Leftarrow$: It is easy to see that coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of B_x is extendible to $B(T_x)$ if there is a coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B_{x'}$ which is extendible to $B(T_{x'})$, where $S \subseteq S'$, $Z \subseteq Z'$ and $A_i \subseteq A'_i$, for all $i \in \{1, \dots, \tau\}$, because $G[B(T_x)]$ is a temporal subgraph of $G[B(T_{x'})]$. Since we want to extend the coloring of B_x such that we have a minimum size S, we take the minimum of all possible colorings $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B_{x'}$ such that $S \subseteq S'$, $Z \subseteq Z'$ and $A_i \subseteq A'_i$, for all $i \in \{1, \dots, \tau\}$.

Note that we can compute the table entry $D_x[A_{[1:\tau]}, S, Z]$ in $\mathcal{O}(|E|)$ time, because we have to look up $\tau + 2$ entries in $D_{x'}$ and $\tau \leqslant |E|$, see Lemmata 2.1 and 2.2. □

Join node. Let $x \in V(T)$ be a join node of $\mathcal{T}$, $x', x'' \in V(T)$ be children of x, and hence $B_x = B_{x'} = B_{x''}$. We set

$$D_x[A_{[1:\tau]}, S, Z] = D_{x'}[A_{[1:\tau]}, S, Z] + D_{x'}[A_{[1:\tau]}, S, Z] - |S|.$$

Lemma 4.13. *Let G be a temporal graph and $\mathcal{T}$ be a tree-decomposition of G as described above, $x \in V(T)$ be a join node of v, $x', x'' \in V(T)$ be the child nodes of x, and $\langle A_{[1:\tau]}, S, Z\rangle$ be a coloring of B_x. Then the following holds:*

(i) Coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of $B_x = B_{x'} = B_{x''}$ is extendible to $B(T_x)$ if and only if it is extendible to $B(T_{x'})$ and $B(T_{x''})$.

(ii) The value of $D_x[A_{[1:\tau]}, S, Z]$ corresponds to Equation (4.1) and can be computed in $\mathcal{O}(1)$ time.

Proof. $\Rightarrow$: Let coloring $\langle A_{[1:\tau]}, S, Z\rangle$ of $B_x = B_{x'} = B_{x''}$ be extendible to $B(T_x)$. Then there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ of $B(T_x)$ such that $S \subseteq S'$, $Z \subseteq Z'$, and $A_i \subseteq A'_i$, for all $i \in \{1, \dots, \tau\}$. Since $B(T_{x'}), B(T_{x''}) \subseteq B(T_x)$ and $B_x = B_{x'} = B_{x''}$, we know that $\langle A_{[1:\tau]}, S, Z\rangle$ is extendible to $B(T_{x'})$ and $B(T_{x''})$.

$\Leftarrow$: Let coloring $\langle A_{[1:\tau]}, S, Z\rangle$ be extendible to $B(T_{x'})$ and $B(T_{x''})$. Assume towards a contradiction that $\langle A_{[1:\tau]}, S, Z\rangle$ is not extendible $B(T_x)$. Thus, there is a valid coloring $\langle A'_{[1:\tau]}, S', Z'\rangle$ for $B(T_{x'})$ such that $S \subseteq S'$, $Z \subseteq Z'$, $A_i \subseteq A'_i$, and there is a coloring $\langle A''_{[1:\tau]}, S'', Z''\rangle$ for $B(T_{x''})$ such that $S \subseteq S''$, $Z \subseteq Z''$, $A_i \subseteq A''_i$, for all $i \in \{1, \dots, \tau\}$. Since $\langle A_{[1:\tau]}, S, Z\rangle$ is not extendible to $B(T_x)$, we know that $\langle A'_1 \cup A''_1, \dots, A'_\tau \cup A''_\tau, S' \cup S'', Z' \cup Z''\rangle$ cannot be a valid coloring. Hence, there is a vertex $v \in B(T_{x''}) \cap B(T_{x'})$ which has different colors in $\langle A'_{[1:\tau]}, S', Z'\rangle$ and $\langle A''_{[1:\tau]}, S'', Z''\rangle$. The set $B^{-1}(v)$ induces a non-empty subtree of T, otherwise $\mathcal{T}$ would not be a tree-decomposition. Therefore, $v \in B_x = B_{x'} = B_{x''}$. This cannot be the case because, then v would have two colors in $\langle A_{[1:\tau]}, S, Z\rangle$. Hence, $\langle A'_1 \cup A''_1, \dots, A'_\tau \cup A''_\tau, S' \cup S'', Z' \cup Z''\rangle$ is a valid coloring of $B(T_x)$, which contradicts $\langle A_{[1:\tau]}, S, Z\rangle$ being not extendible to $B(T_x)$.

Furthermore, this implies that for all vertices $w \in B(T_x)$ it holds that $w \in S' \cap S''$ implies $w \in S$. Hence, $|S'| + |S''| - |S| = |S'| + |S''| - |S \cap S''| = |S' \cup S''|$.

Note that we can compute the table entry $D_x[A_{[1:\tau]}, S, Z]$ in $\mathcal{O}(1)$ time, because we just have to look up one table entry of $D_{x'}$ and one in $D_{x''}$. □

Having shown the previews Lemmata 4.7 to 4.13, we are now all set to prove Theorem 4.6.

Proof of Theorem 4.6. First, we show an algorithm for Non-Strict (s,z)-Separation and conclude afterwards that the algorithm can be adapted to Strict (s,z)-Separation. The algorithm works as follows on the Non-Strict (s,z)-Separation input instance $I = (G = (V, E, \tau), s, z, k)$.

(i) Compute a nice tree-decomposition $\mathcal{T}$ for the underlying graph $G_\downarrow$ in $2^{\mathcal{O}(\mathrm{tw}(G_\downarrow)^3)} \cdot |V|$ time.

(ii) Add s and z to every bag in $\mathcal{O}(\mathrm{tw}(G_\downarrow)\cdot|V|)$ time. Note that $|V(T)| \in \mathcal{O}(\mathrm{tw}(G_\downarrow)\cdot|V|)$ and that each bag is of size at most $\mathrm{tw}(G_\downarrow) + 2$.

(iii) Compute the dynamic program of Equation (4.1) on $\mathcal{T}$. This can be done in $\mathcal{O}((\tau+2)^{\mathrm{tw}(G_\downarrow)+2} \cdot \mathrm{tw}(G_\downarrow) \cdot |V| \cdot |E|)$, because there are at most $(\tau+2)^{\mathrm{tw}(G_\downarrow)+2}$ possible colorings for each bag, there are at most $\mathcal{O}(\mathrm{tw}(G_\downarrow) \cdot |V|)$ many bags, and table entry for one coloring can be computed in $\mathcal{O}(|E|)$ time, see Lemmata 4.8 to 4.13.

(iv) Iterate over the root table D_r. If there is an entry of size at most k, then output yes, otherwise output no. The correctness of this step follows from Lemma 4.7.

Hence, the input instance I can be decided in

$$\mathcal{O}(\underbrace{(\tau+2)^{\mathcal{O}(\mathrm{tw}(G_\downarrow)^3)} \cdot \mathrm{tw}(G_\downarrow)}_{=:f(\tau,\mathrm{tw}(G_\downarrow))} \cdot |V| \cdot |E|) \text{ time.}$$

Now let $I_1 = (G = (V, E, \tau), s, z, k)$ be a Strict (s,z)-Separation instance. By the reduction of Corollary 3.2 we compute an equivalent Non-Strict (s,z)-Separation instance $I_2 = (G' = (V', E', 2\tau), s, z, k)$ such that $|V'| = |V| + 2 \cdot |E|$ and $|E'| = 4 \cdot |E|$. The underlying treewidth $\mathrm{tw}(G'_\downarrow)$ of G' can be upper bounded by $\mathrm{tw}(G_\downarrow) + 2 \cdot \binom{\mathrm{tw}(G_\downarrow)}{2} \cdot \tau$, because there are at most $\binom{\mathrm{tw}(G_\downarrow)}{2} \cdot \tau$ time-edges in the temporal subgraph of a bag of a tree-decomposition of $G_\downarrow$. Hence, the input instance I_1 can be decided in $\mathcal{O}(f(2\tau, \mathrm{tw}(G_\downarrow) + 2 \cdot \binom{\mathrm{tw}(G_\downarrow)}{2} \cdot \tau) \cdot |V| \cdot |E| + |E|^2)$ time. □

In order to answer the question from Section 3.3 whether Non-Strict (s,z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ, one could ask whether the underlying treewidth of a temporal graph can be upper-bounded by a computable function f in k and τ. In general, the treewidth of a graph and the minimum size of a (s,z)-separator are incomparable. We refer to Figure 4.3 for an example.

This implies that such a function f does not exist. However, the *treewidth reduction technique* of Marx, O'sullivan, and Razgon [MOR13] modifies a graph such that we can upper-bound the treewidth by a function in k. It is not clear whether this technique can be lifted to temporal graphs.

A natural question is whether we can drop one of the parameters from Theorem 4.6. From Theorem 3.4, we know that (Non-)Strict (s,z)-Separation is not fixed-parameter tractable when parameterized by the maximum label τ, unless P = NP.

(a) $\mathrm{tw}(G_1) > k$

(b) $k > \mathrm{tw}(G_2)$

Figure 4.3: Figure 4.3a shows a graph G_1 in which $\{v\}$ is an (s, z)-separator of size one, but the treewidth of G_1 depends on the size of the clique in G_1. Figure 4.3b shows a graph G_2 where each vertex, except s and z, must be in an (s, z)-separator, but the treewidth of G_2 is at most two. To see this, we can construct a tree-decomposition of G_2 by putting each vertex together with s, z in one bag of size three and then create a root bag which contains s, z. Finally, we connect each non-root bag with the root bag. That is a tree-decomposition for G_2 of width two.

Now, we are going to discuss whether the (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the underlying treewidth, without fully answering this question.

On the one hand, a naive dynamic program for a problem with connectivity constraints on a graph G keeps track of every path in G. On a graph of bounded treewidth, this leads to a running time of $\mathrm{tw}(G)^{\mathcal{O}(\mathrm{tw}(G))} \cdot |V(G)|^{\mathcal{O}(1)}$ for many problems. The *Cut&Count* technique of Cygan et al. [Cyg+11] is a tool to improve those dynamic programs to a running time of $c^{\mathrm{tw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, where c is a small constant. The idea behind *Cut&Count* seems to be promising to improve the running time of the dynamic program of Theorem 4.6 to $f(\mathrm{tw}_{\downarrow}(G)) \cdot (|V| \cdot |E|)^{\mathcal{O}(1)}$, where f is a computable function.

On the other hand, we there is evidence that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by the underlying treewidth, which we now will try to explain. Note that the evidence that (Non-)Strict (s, z)-Separation in W[1]-hard is rather weak.

A *path-decomposition* of a graph G is a tree-decomposition $(T, (B_i)_{i \in V(T)})$, where T is a path and the *pathwidth* $\mathrm{pw}(G)$ of G is defined as the minimal width over all path-decompositions of G. Hence, the treewidth of a graph can be upper-bounded by the pathwidth.

Dvořák and Knop [DK15] proved that Length-Bounded (s, z)-Cut is W[1]-hard with respect to the pathwidth, and hence treewidth, of the input graph. Length-Bounded (s, z)-Cut and Length-Bounded (s, z)-Separation usually behave similarly in computational complexity [Bai+10; Flu+16; GT11]. However, to the best of our knowledge, it is open whether Length-Bounded (s, z)-Separation is W[1]-hard when parameterized by the treewidth of the input graph. If Length-Bounded (s, z)-Separation is W[1]-hard when parameterized by the treewidth of the input graph, then (Non-)Strict (s, z)-Separation is also W[1]-hard when parameterized by underlying treewidth, by the reductions of Theorem 3.1 and Corollary 3.2. Note that in the polynomial reduction of Golovach and Thilikos [GT11] from an instance (G, s, z, k, ℓ) of Length-Bounded (s, z)-Cut to an instance $(G', s', z', k, \ell + 1)$ of Length-Bounded (s, z)-Separation, the graph G' is the line graph of G plus two vertices.

The *line graph* of an graph G is a graph G' that represents the adjacencies between edges of G. To the best of our knowledge, there is no way to upper-bound the pathwidth (or treewidth) of G' by the pathwidth (or treewidth) of G. For a precise definition of a line graph and further discussions about the treewidth of line graphs, we refer to Harvey [Har14].

Cliquewidth is a similar graph parameter to treewidth, for which we do not want to give a precise definition. Instead, we refer to Courcelle and Engelfriet [CE12]. Roughly speaking, there is an efficient algorithm for a problem on graphs of bounded treewidth or cliquewidth, if it is definable in monadic second-order logic [CE12]. Note that Section 4.1.2 is about definability of temporal graph problems in monadic second-order logic. Furthermore, we can upper bound the treewidth of a graph by its cliquewidth [CR01]. Hence, if a graph problem is fixed-parameter tractable with respect to the treewidth, then it is also fixed-parameter tractable with respect to the cliquewidth.

Corollary 4.14. Strict (s, z)-Separation *is* W[1]*-hard when parameterized by the cliquewidth of the underlying graph.*

Proof. Let $I_1 = (G, s, z, k, \ell)$ be a Length-Bounded (s, z)-Cut instance, where G has pathwidth $\mathrm{pw}(G)$. We compute the Length-Bounded (s, z)-Separation instance $I_2 = (G', s', z', k, \ell + 1)$ with the polynomial reduction of Golovach and Thilikos [GT11]. Note that $G' - \{s', z'\}$ is the line graph of G. This implies that the cliquewidth of $G' - \{s', z'\}$ can be upper-bounded by a computable function in $f(\mathrm{pw}(G))$ [GW07], and therefore Length-Bounded (s, z)-Separation is W[1]-hard with respect to the cliquewidth. We construct the Strict (s, z)-Separation instance $\hat{O} := (\hat{G}, \hat{s}, \hat{z}, k)$ from I_2 with the reduction of Theorem 3.1. Note that the vertex sets of G' and $\hat{G}'$ are equal and that $(e, t) \in E(\hat{G}) \Leftrightarrow e \in E(G')$. Consequently, G' is the underlying graph of $\hat{G}$. Thus, the cliquewidth of the underlying graph of $\hat{G}$ is also upper-bounded by $f(\mathrm{pw}(G))$. Hence, Strict (s, z)-Separation is W[1]-hard when parameterized by the cliquewidth of the underlying graph. □

It seems that, with little effort, Corollary 4.14 could be extended to Non-Strict (s, z)-Separation using the reduction from Corollary 3.2. However, there are problems which are W[1]-hard when parameterized by cliquewidth, but fixed-parameter tractable, when parameterized by treewidth [Fom+10]. Thus, (Non-)Strict (s, z)-Separation can be fixed-parameter tractable as well.

4.1.2 Monadic Second-Order Logic on Temporal Graphs

In this section, we transfer a powerful tool of Mans and Mathieson [MM14] from dynamic graphs to temporal graphs, which shows fixed-parameter tractability by expressing the problem in monadic second-order (MSO) logic. Afterwards, we use this tool to show that Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k, the maximum label τ and the layer treewidth $\mathrm{tw}_{\max}$. Finally, we show that there is (presumably) no hope that (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by

(i) the maximum label τ and the layer treewidth $\mathrm{tw}_{\max}$, or

(ii) the solution size k and the layer treewidth $\mathrm{tw}_{\max}$.

Mans and Mathieson [MM14] studied the treewidth of dynamic graphs. In their model, time is not necessarily linearly ordered and edges as well as vertices can appear and disappear over time. Thus, their model is more powerful than temporal graphs. For example, they can easily model scenarios on a multi-core processor, where time is partially ordered. However, they also looked into dynamic graphs where the time is linearly ordered and showed that there are logical structures for temporal graphs where the layer treewidth can be preserved.

A structure $\mathcal{A}$ consists of a finite set A of elements, called universe, and a non-empty set Φ of relations of finite arity called the vocabulary, along with an interpretation of each relation in Φ over A. To be precise, for each relation $R \in \Phi$ of arity r there is a set of tuples from A^r that define when R is true.

Let $G = (V, E, \tau)$ be a temporal graph. We define the structure $A(G) := (A, \Phi)$ obtained from G with universe A and vocabulary Φ, where

- $\mathcal{A} := \{v^t \mid v \in V \text{ and } t \in \{1, \ldots, \tau\}\} \cup \{t \mid t \in \{1, \ldots, \tau\}\} \cup \{b\}$ equipped with a function f_V such that $f_V(v^t) = f_V(u^{t'})$ if and only if $v^t, u^{t'}$ are derived from the same $v \in V$, and
- $\Phi := \{\mathcal{V}, \mathcal{E}, T, R\}$ such that
  - $v^t \in \mathcal{V}$ if and only if $v \in V$ and $t \in \{1, \ldots, \tau\}$,
  - $(v^t, w^t, t) \in \mathcal{E}$ if and only if $(\{v, w\}, t) \in E$,
  - $t \in T$ if and only if $t \in \{1, \ldots, \tau\}$,
  - $(t_1, t_2) \in R$ if and only if $t_1, t_2 \in \{1, \ldots, \tau\}$ and $t_1 + 1 = t_2$, and
  - $(b, t_i) \in R$ if and only if $t_i = \min\{1, \ldots, \tau\} = 1$.

The relation $\mathcal{V}$ describes whether an element is a vertex of the temporal graph, the relation $\mathcal{E}$ describes whether there is a time-edge between these vertices at a specific time, the relation T describes if an element is a time point, and the relation R describes the linear order of the time points, where b is before the first time point.

Given a structure, the *Gaifman graph* is obtained by taking the universe of the logical structure as the vertex set, with an edge between two vertices if they ever appear in the interpretation of any relation in the vocabulary together.

Theorem 4.15 (Mans and Mathieson [MM14])**.** *Let G be a temporal graph and $\mathcal{A}(G)$ the logical structure obtained by G, and $\mathrm{tw}_{\max} := \max_{i \in \{1,\ldots,\tau\}} \mathrm{tw}(G_i)$ the layer treewidth of G. Then the treewidth of the Gaifman graph is at most $\mathrm{tw}_{\max} + 1$.*

Figure 4.4 shows a Gaifman graph of a temporal graph. The tree-decomposition of a Gaifman graph $\mathcal{G}$ of a structure $\mathcal{A}(G)$ obtained from a temporal graph $G = (V, E, \tau)$ is constructed as follows: Let $\omega = \max_{i \in \{1,\ldots,\tau\}} \mathrm{tw}(G_i)$ be the layer treewidth of G. Hence, for each layer i of G there is a tree-decomposition of width at most ω. Note that the elements of T from $\mathcal{A}(G)$ form a path in the Gaifman graph. Thus, there is a tree-decomposition of T of width one. Let $t_i \in T$. Now, we pick a bag B_x which contains t_i, add t_i to every bag in the tree-decomposition of G_i and connect B_x to an arbitrary bag of the tree-decomposition of G_i. This gives us a tree-decomposition of $\mathcal{G}$ of width $\omega + 1$.

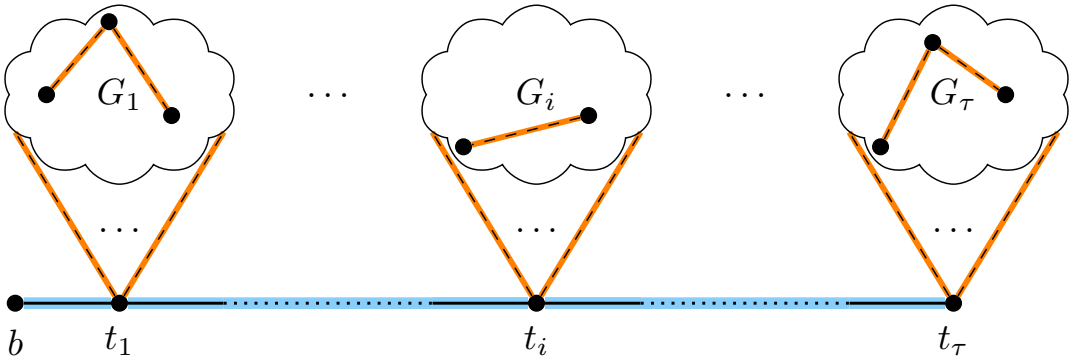


Figure 4.4: The Gaifman graph of a structure $\mathcal{A}(G)$ obtained from a temporal graph G. The edges corresponding to the relation R are blue (solid) and the edges corresponding to the relation $\mathcal{E}$ are orange (dashed).

Monadic second-order (MSO) logic consists of a countably infinite set of (individual) variables, unary relation variables, and a vocabulary Φ. We assume that Φ matches the vocabulary of the structure the logic is being applied to. A *formula* in monadic second-order logic is constructed from the variables, vocabulary, the relations $=, \wedge, \vee, \neg$, and the quantifiers $\forall$ and $\exists$. A *sentence* is a formula where all variables are bounded by a quantifier. We omit the precise semantic of how an MSO sentence is applied to a structure and refer to Ebbinghaus and Flum [EF05] and Ebbinghaus, Flum, and Thomas [EFT94] for further information, as Mans and Mathieson [MM14] have done. Mans and Mathieson [MM14] stated that Courcelle [Cou06; Cou90] have shown the following theorem in a series of papers (see also Courcelle and Engelfriet [CE12]).

Theorem 4.16 (Courcelle and Engelfriet [CE12])**.** *Let $\mathcal{A}(G)$ be the structure obtained from a temporal graph G, and $\mathcal{G}$ be the Gaifman graph of $\mathcal{A}(G)$. The problem whether the structure $\mathcal{A}(G)$ satisfies an MSO sentence ρ is fixed-parameter tractable when parameterized by the treewidth of the Gaifman graph $\mathrm{tw}(\mathcal{G})$ and the length of the MSO sentence $|\rho|$.*

Theorem 4.16 together with Theorem 4.15 gives us a powerful tool to show fixed-parameter tractability for problems on temporal graphs. But admittedly these algorithms are far from practical, because the running time can contain a tetration in the power of the parameter.

However, we are able to show fixed-parameter tractability of Non-Strict (s, z)-Separation when parameterized by the solution size, the maximum label, and the layer treewidth. Note that the following approach also works for Strict (s, z)-Separation, but Theorem 3.15 already gives us a faster algorithm.

Theorem 4.17. Non-Strict (s, z)-Separation *is fixed-parameter tractable when parameterized by the solution size k, the maximum label τ, and the layer treewidth $\mathrm{tw}_{\max} = \max_{i \in \{1,\dots,\tau\}} \mathrm{tw}(G_i)$, where G_i is layer i of the temporal input graph.*

Proof. We will show that Non-Strict (s, z)-Separation is expressible in an MSO sentence for the structure obtained from the temporal input graph, and that the length

of the MSO sentence can be upper bounded by the solution size k and the maximum label τ.

Let $G = (V, E, \tau)$ be a temporal graph, $s, z \in V$ two distinct vertices and $k \in \mathbb{N}$ be an integer. One can observe that the structure $\mathcal{A}(G)$ and its Gaifman graph can be constructed in polynomial-time.

We ask for elements $s_1, \ldots, s_k$ and $t_1, \ldots, t_\tau$ such that $\mathcal{V}(s_i) = \text{true}$, $T(t_j) = \text{true}$, and $t_1, \ldots, t_\tau$ are ordered by the distance to b in the Gaifman graph, where $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, \tau\}$. That is, $s_1, \ldots, s_k$ are vertices and $t_1, \ldots, t_\tau$ are time points in G. In the end, $\{s_1, \ldots, s_k\}$ will be the separator. That the distance of t_i to b is smaller than the distance of t_{i+1} to b can be expressed by the recursive MSO sentence [MM14]

$$t_i < t_{i+1} := \bigvee_{\ell=1}^{\tau} d_\ell(t_i) \wedge \neg d_\ell(t_{i+1}), \tag{4.2}$$

where $i \in \{1, \ldots, \tau - 1\}$ and

$$\begin{aligned} d_1(t_i) &= R(s, t_i), \\ d_n(t_i) &= \exists t \left[T(t) \wedge R(t, t_i) \wedge d_{n-1}(t) \right]. \end{aligned}$$

The recursion depth of $d_\tau(t_i)$, and therefore the length of the MSO sentence in Equation (4.2) is bounded in τ.

For each $t_i \in \{t_1, \ldots, t_\tau\}$, we ask for a non-empty connected component P_i of vertices, $\forall v \in P : \mathcal{V}(v) = \text{true}$, such that all vertices in P_i are connected to t_i, $\forall v \in P_i : \mathcal{E}(v, t_i)$. One can observe that P_i is a connected component of G_i. Connectivity of a vertex set can be expressed in an MSO sentence by requesting an edge between all possible partitions of the vertex set into two vertex sets.

$$\text{CONN}(P) := \forall S \subseteq P \left[(\forall x (S(x)) \vee \forall y (\neg S(y))) \vee \exists x, y \left(S(x) \wedge S(y) \wedge \mathcal{E}(x, y) \right) \right]$$

To get a non-strict (s, z)-path we assure that P_1 contains s^1, P_τ contains z^τ, and two consecutive P_i, P_{i+1} have a vertex in common under the function f_V, where $i \in \{1, \ldots, \tau - 1\}$. Note that P_i can consist of just one vertex and that $P_1, \ldots, P_\tau$ may not exactly correspond to the visited vertices of a non-strict (s, z)-path, but there are subsets $P'_1 \subseteq P_1, \ldots, P'_\tau \subseteq P_\tau$ such that there is a non-strict (s, z)-path which exactly visits the vertices $\{f_V(v) \mid v \in P'_1 \cup \ldots, \cup P'_\tau\}$.

To separate s from z, we want that one of the vertex sets $P_1, \ldots, P_\tau$ contains at least

one vertex of $s_1, \ldots, s_k$ under the f_V function. We get the following MSO sentence $\rho :=$

$$
\begin{aligned}
&\exists s_1, \ldots, s_k, \forall t_1, \ldots, t_\tau, P_1, \ldots, P_\tau(\\
&\quad \bigwedge_{i=1}^{\tau-1} t_i < t_{i+1} \wedge \\
&\quad \bigwedge_{i=1}^{\tau} \forall v(P_i(v) \wedge \mathcal{V}(v) \wedge \mathcal{E}(t_i, v)) \wedge \mathrm{CONN}(P_i) \\
&\quad \wedge P_1(s^1) \wedge P_\tau(z^\tau) \wedge \\
&\quad \bigwedge_{i=1}^{\tau-1} \exists x, y \, (P_i(x) \wedge P_{i+1}(y) \wedge f_V(x) = f_V(y)) \\
&\quad \wedge \exists w(\\
&\qquad \bigvee_{i=1}^{\tau} \left[P_i(w) \wedge \left(\bigvee_{j=1}^{k} f_V(w) = f_V(s_j) \right) \right] \\
&\quad) \\
&)
\end{aligned}
\tag{4.3}
$$

One can observe that the length of ρ is upper-bounded by a computable function in $k+\tau$.

We claim that there is a non-strict (s, z)-separator S of size at most k in G if and only if ρ is satisfied.

$\Rightarrow$: Let $S = \{v_1, \ldots, v_k\}$ be a non-strict (s, z)-separator of size at most k in G. We assign $s_1 = v_1^1, \ldots, s_k = v_k^1$. Assume towards a contradiction that there are $t_1, \ldots, t_\tau$ and $P_1, \ldots, P_\tau$ from $\mathcal{A}(G)$ such that

$$
\begin{aligned}
&\bigwedge_{i=1}^{\tau-1} t_i < t_{i+1} \\
&\bigwedge_{i=1}^{\tau} \forall v(P_i(v) \wedge \mathcal{V}(v) \wedge \mathcal{E}(t_i, v)) \wedge \mathrm{CONN}(P_i) \\
&\wedge P_1(s^1) \wedge P_\tau(z^\tau) \\
&\bigwedge_{i=1}^{\tau-1} \exists x, y \, (P_i(x) \wedge P_{i+1}(y) \wedge f_V(x) = f_V(y)) \, ,
\end{aligned}
\tag{4.4}
$$

and let $w_0, \ldots, w_\tau \in V$ such that $w_0 = s$, $w_\tau = z$, and $w_i^i \in P_i$ and $w_i^{i+1} \in P_{i+1}$, for $i \in \{1, \ldots, \tau-1\}$. Then, we know that P_i contains just one vertex or there is a subset $P_i' \subset P_i$ such that there is a path W_i from w_{i-1} to w_i in G_i such that $V(W_i) = P_i'$, for $i \in \{1, \ldots, \tau\}$. If $|P_i| = 1$, then we set $P_i' = P_i$, for all $i \in \{1, \ldots, \tau\}$. Clearly, we can obtain a non-strict (s, z)-path X from $P_{i_1}', \ldots, P_{i_j}'$. Since S is a non-strict (s, z)-separator, we know that there is a $v_\ell \in S \cap V(X)$. Hence, there is an $i \in \{1, \ldots, \tau\}$ such that $v_\ell^i \in P_i' \subseteq P_i$ and therefore $f_V(v_\ell^i) = f_V(v_\ell^1) = f_V(s_\ell)$. This is a contradiction and implies that ρ is satisfiable.

$\Leftarrow$: Let ρ be satisfied. We set $S := \{f_V(s_1), \ldots, f_V(s_k)\}$. Note that $|S| \leqslant k$. Assume towards a contradiction that there is a non-strict (s, z)-path P in $G - S$. Now, we construct a sequence of sets $P_1, \ldots, P_\tau$ such that $P_t := \{v^t \mid (e, t) \in E \text{ and } v \in e\}$ and s^1 to P_1 and z^τ to P_τ, where $t \in \{1, \ldots, \tau\}$. Some of the sets might be empty.

Let $P_t = \emptyset$, $t_1 = \arg\max_{i\in\{1,\dots,t-1\}} P_i \neq \emptyset$, and $t_2 = \arg\min_{i\in\{t+1,\dots,\tau\}} P_i \neq \emptyset$, for some $t \in \{1,\dots,\tau\}$. Since P is a non-strict (s,z)-path, we know that there is a $w \in V(P)$ such that $w^{t_1} \in P_{t_1}$ and $w^{t_2} \in P_{t_2}$. We add $w^{t'}$ to $P_{t'}$, for all $t' \in \{t_1+1,\dots,t_2-1\}$. The sentence ρ is satisfied implies that there is $j \in \{1,\dots,k\}$ and a $w^t \in P_t$ such that $f_V(w^t) = f_V(s_j)$. This contradicts P being a non-strict (s,z)-path in $G-S$, because $w \in S$.

Hence Non-Strict (s,z)-Separation is expressible in an MSO sentence, where the length of the sentence can be upper bounded in $k+\tau$. By Theorems 4.15 and 4.16, this shows fixed-parameter tractability of Non-Strict (s,z)-Separation when parameterized by $k+\tau+\mathrm{tw}_{\max}$, where $\mathrm{tw}_{\max} = \max_{i\in\{1,\dots,\tau\}} \mathrm{tw}(G_i)$ and G_i is the layer i of G. □

A natural question is whether we can drop one of the parameters from Theorem 4.17. In the following theorem we are going to show that we cannot drop k, unless NP = P.

Corollary 4.18. Strict (s,z)-Separation *is* NP*-hard, even if* $\tau = 6$ *and* $\mathrm{tw}_{\max} = 1$ *and* Non-Strict (s,z)-Separation *is* NP*-hard, even if* $\tau = 4$ *and* $\mathrm{tw}_{\max} = 1$ *, where* $\mathrm{tw}_{\max} = \max_{i\in\{1,\dots,\tau\}} \mathrm{tw}(G_i)$ *and* G_i *is the layer the temporal input graph.*

Proof. We say a graph is *k-regular* if all vertices have degree exactly k. We will reduce, similarly to Theorem 3.4, from Vertex Cover on 3-regular graphs. For this, we need to introduce the very similar problem of finding an Independent Set in a graph.

Independent Set
Input: A graph $G = (V,E)$ and an integer k.
Question: Is there a subset $V' \subseteq V$ of size at most k such that there is no edge between two vertices in V'?

Fricke, Hedetniemi, and Jacobs [FHJ98] showed that Independent Set is NP-hard on 3-regular graphs, and therefore, Vertex Cover is NP-hard on 3-regular graphs, because a graph with n vertices has an Independent Set of size k if and only if it has a Vertex Cover of size $n-k$ [Kar72].

Let $(G=(V,E),k)$ be a Vertex Cover instance, where G is 3-regular and $|V| = n$. We call an edge coloring $f : E \to \{1,\dots,h\}$ of h colors *valid* if for all incident edges $e_1, e_2 \in E$ it holds that $f(e_1) \neq f(e_2)$. By an algorithm of Misra and Gries [MG92] for Vizing's Theorem, we can compute a valid edge-coloring with at most $\Delta+1$ colors in polynomial-time. Since G has maximum degree $\Delta = 3$, we can compute a valid edge-coloring $f : E \to \{1,\dots,4\}$ with four colors for G in polynomial-time.

We construct a Non-Strict (s,z)-Separation instance $\hat{O} := (\hat{G} = (\hat{V}, \hat{E}, \tau = 4), s, z, k+n)$, where

$$\hat{V} := V \cup \{v_1, v_2 : v \in V\} \cup \{s,z\}$$

are the vertices and the time-edges are defined as

$$\hat{E} := \overbrace{\{(\{s,v_1\},1),(\{v_1,v\},1),(\{v,v_2\},3),(\{v_2,z\},4),(\{s,v\},3),(\{v,z\},2) : v \in V\}}^{\text{vertex-edges}} \cup \underbrace{\{(\{v_1,w_2\},f(\{v,w\})),(\{w_1,v_2\},f(\{v,w\})) : \{v,w\} \in E\}}_{\text{edge-edges}}.$$

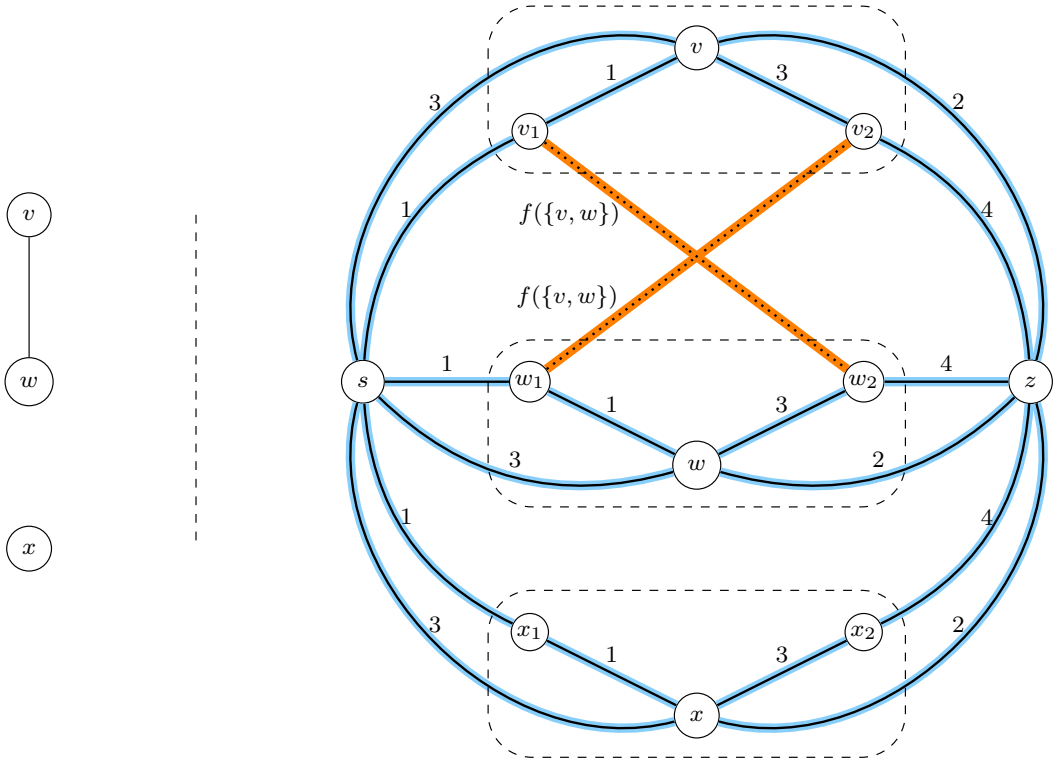


Figure 4.5: The VERTEX COVER instance $(G, 1)$ (left) and the corresponding NON-STRICT (s, z)-SEPARATION instance from the reduction of Corollary 4.18 (right). The edge-edges are orange (dotted), the vertex-edges are blue (solid), and the vertex gadgets are in dashed boxes. The function f is an edge coloring of the input graph G.

Observe that $|\hat{V}| = 3 \cdot n + 2$, $|\hat{E}| = 6 \cdot |\hat{V}| + 2 \cdot |E|$, $\tau = 4$, and that $\hat{G}$ can be computed in polynomial time. To see that there is a vertex cover for G of size at most k if and only if there is a non-strict (s, z)-separator in $\hat{G}$ of size at most $n + k$, we refer to the proof of Theorem 3.4.

The label of an edge-edge for $\{v, w\} \in E$ depends on the color of $\{v, w\}$ under f. Since all incident edges are of different colors, there is no layer where two edge-edges are incident to each other. One can observe that each layer of $\hat{G}$ is a forest and therefore has treewidth one. Consequently, $\mathrm{tw}_{\max} = \max_{i \in \{1,\dots,\tau\}} \mathrm{tw}(G_i) = 1$ and NON-STRICT (s, z)-SEPARATION is NP-hard, even if $\tau = 4$ and $\mathrm{tw}_{\max} = 1$.

With one small adjustment, we can reduce from VERTEX COVER on 3-regular graphs to STRICT (s, z)-SEPARATION. Here, the time-edges are defined as

$$\hat{E} := \overbrace{\{(\{s, v_1\}, 1), (\{v_1, v\}, 2), (\{v, v_2\}, 5), (\{v_2, z\}, 6), (\{s, v\}, 4), (\{v, z\}, 3) : v \in V\}}^{\text{vertex-edges}} \cup \underbrace{\{(\{v_1, w_2\}, f(\{v, w\}) + 1), (\{w_1, v_2\}, f(\{v, w\}) + 1) : \{v, w\} \in E\}}_{\text{edge-edges}}.$$

Hence, STRICT (s, z)-SEPARATION is NP-hard, even if $\tau = 6$ and $\mathrm{tw}_{\max} = 1$. □

Furthermore, we can observe that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size k and the layer treewidth $\mathrm{tw}_{\max}$. Hence, we (presumably) cannot drop the parameter τ from Theorem 4.17.

Corollary 4.19. (Non-)Strict (s, z)-Separation *is* W[1]*-hard when parameterized by the solution size* k*, even if the layer treewidth* $\mathrm{tw}_{\max}$ *is one.*

Proof. Let $I := (G = (V, E, \tau), s, z, k)$ be an instance of Strict (s, z)-Separation. From Theorem 3.1, we already know that Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size.

Now, we construct an instance $O_1 := (G' = (V', E', 2 \cdot \tau), s, z, k)$ of Non-Strict (s, z)-Separation by the reduction of Corollary 3.2. Recall from the proof of Corollary 3.2, that there is a strict (s, z)-separator in G if and only if there is non-strict (s, z)-separator in G'. Hence, the correctness of this reduction is already shown.

Let $V_e \subseteq V'$ be the set of all edge-vertices in V'. Note that each vertex $v \in V_e$ has degree at most one in layer i of G', where $i \in \{1, \ldots, 2 \cdot \tau\}$. Furthermore, there are no two non-edge-vertices $w, w' \in V' \setminus V_e$ such that there is a time-edge between w and w'. Thus, each layer of G' is a disjoint union of stars. A graph is called *star* if it is connected and has only one vertex of degree more than one. Since a star is a tree, we have treewidth at most one for each layer in G'. Hence, we can upper-bound $k + \mathrm{tw}_{\max}$ by $k + 1$, where $\mathrm{tw}_{\max} := \max_{i \in \{1,\ldots,2\tau\}} \mathrm{tw}(G'_i)$. This already shows that Non-Strict (s, z)-Separation is W[1]-hard when parameterized by the solution size k and the layer treewidth $\mathrm{tw}_{\max}$ of temporal input graph.

Now, we construct an instance $O_2 = (G', s, z, k)$ of Strict (s, z)-Separation. Note that O_1 and O_2 have the same temporal graph G'. Since each non-strict (s, z)-separator is also a strict (s, z)-separator in a temporal graph, we know that there is a strict (s, z)-separator in G if and only if there is a strict (s, z)-separator in G'. Hence, I is a yes-instance if and only if O_2 is a yes-instance, which completes this proof. □

From the reduction of Corollary 4.19 one can observe that in the strict path model we can always construct a temporal graph where the layer treewidth is one without creating or destroying any strict (s, z)-path. Hence, for many problems in the strict path model we can show fixed-parameter tractablility by the length of the MSO sentence of the problem. However, we already showed by Theorem 3.15 that Strict (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ.

Now, we discuss whether one can drop the parameter $\mathrm{tw}_{\max}$ from Theorem 4.17. One can observe that this would answer the open question from the end of Section 3.3 whether Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by solution size k and maximum label τ. We will not settle this question in this thesis.

The *treewidth reduction* technique of Marx, O'sullivan, and Razgon [MOR13] seems to be a promising approach to upper bound $\mathrm{tw}_{\max}$ by k, but unfortunately we could not bring this idea to work without introducing too many elements to the universe of $\mathcal{A}(G)$ of Theorem 4.17. In the treewidth reduction technique, one constructs a helper graph G' for the input graph G by, roughly speaking, contracting all parts of G which cannot help to find an (s, z)-separator. Thus, every (s, z)-separator of G is also present in G'.

The main difficulty of using this idea in the Gaifman graph seems to be that a non-strict (s, z)-path could start to use edges from another layer in those parts of the current layer which are contracted by the treewidth reduction technique.

Nevertheless, if we consider a multilayer setting, then the treewidth reduction technique is a helpful tool. We ask for a set $S \subseteq V$ of size at most k in a temporal graph $G = (V, E, \tau)$ such that there is no (s, z)-path in $G_i - S$ for any layer i of G, where $s, z \in V$. One can observe that we ask for a 0-bounded (s, z)-separator. Now, we can run the treewidth reduction technique on the induced subgraph of the Gaifman graph which represents the layer i of the temporal graph. Hence, we conjecture that 0-Bounded (s, z)-Separation is fixed-parameter tractable when parameterized by the solution size k and the maximum label τ.

4.2 Temporal Planar Graphs

In Chapter 1, we noticed that we can model public transport systems in temporal graphs. Some of these systems, like bus and train systems, operate often on so-called planar graphs. Hence, it is of interest to determine the computational complexity of (Non-) Strict (s, z)-Separation on the class of temporal planar graphs.

In this section, we give a definition of temporal planar graphs, show that (Non-) Strict (s, z)-Separation is NP-hard on temporal planar graphs, and give an FPT-algorithm with respect to τ for Strict (s, z)-Separation on classes of underlying graphs of bounded local treewidth. This algorithm applies also to Strict (s, z)-Separation on temporal planar graphs.

A *polygonal line* is a subset $X \subset \mathbb{R}^2$ such that there is a homeomorphism $h : [0, 1]_{\mathbb{R}} \to X$, where $[0, 1]_{\mathbb{R}}$ is the closed interval from 0 to 1 in $\mathbb{R}$. We can think of this as a finite line in the plane. A graph $G = (V, E)$ is *planar* if there are functions $f_V : V \to \mathbb{R}^2$ and $f_E : E \to 2^{\mathbb{R}^2}$ such that

(i) for all distinct vertices $v, w \in V$ it holds that $f_V(v) \neq f_V(w)$,

(ii) for all edges $\{v, w\} \in E$ the set $f_E(\{v, w\})$ is a polygonal line for which there exists a homeomorphism $h : [0, 1]_{\mathbb{R}} \to f_E(\{v, w\})$ such that $h(0) = f_V(v)$ and $h(1) = f_V(w)$,

(iii) for all distinct edges $e_1, e_2 \in E$ with $e_1 \cap e_2 = \emptyset$ it holds that $f_E(e_1) \cap f_E(e_2) = \emptyset$, and

(iv) for all distinct edges $e_1, e_2 \in E$ with $e_1 \cap e_2 = \{v\}$ it holds that $f_E(e_1) \cap f_E(e_2) = \{f_V(v)\}$.

The function set $\{f_V, f_E\}$ is called an *embedding* of G in the plane. We say a temporal graph is a *temporal planar graph* if its underlying graph is planar.

Fluschnik et al. [Flu+16] showed that Length-Bounded (s, z)-Cut is NP-hard on planar graphs by a reduction from Vertex Cover on planar graphs with maximum degree $\Delta = 3$. By examining the proof of this result, one can observe that instance of Length-Bounded (s, z)-Cut has maximum degree $\Delta = 6$.

Lemma 4.20 (Fluschnik et al. [Flu+16])**.** Length-Bounded (s,z)-Cut *is* NP*-hard on planar graph with maximum degree* $\Delta = 6$ *and the degrees of the special vertices* s, z *are three.*

First, we show that Length-Bounded (s,z)-Separation is NP-hard on planar graphs. Second, we will use the reductions from Theorem 3.1 and Corollary 3.2 to show that (Non-)Strict (s,z)-Separation in NP-hard on temporal planar graphs.

Lemma 4.21. Length-Bounded (s,z)-Separation *is* NP*-hard on planar graphs.*

Proof. Let $I := (G = (V,E), s, z, \ell, k)$ be an instance of Length-Bounded (s,z)-Cut, where G is planar, has maximum degree $\Delta = 6$, and the degree of s and z is three. It is NP-hard to decided whether I is a yes-instance, see Lemma 4.20. We will now provide a reduction from Length-Bounded (s,z)-Cut to Length-Bounded (s,z)-Separation.

Construction. We construct an instance $O := (G', s', z', \ell', k)$ of Length-Bounded (s,z)-Separation as follows.

For a vertex $v \in V$, we introduce a *vertex-gadget* G_v which is a grid of size $(2k+1) \times (2k+1)$. There are six pairwise disjoint *connector sets* $C_{(v,1)}, \dots, C_{(v,6)} \subseteq V(G_v)$ of size $k+1$. Two above of G_v, two below of G_v, one on the left side of G_v, and one on the right side of G_v. See Figure 4.6 for an example.

One can observe that all cycle-free (x,y)-paths are of length at most $k' := (2k+2)^2 - 1$, for all $x, y \in V(G_v)$, because there are only $(2k+2)^2$ vertices in $V(G_v)$. For each vertex $v \in V$, we add such a vertex-gadget G_v to G'.

Without loss of generality, we assume that the neighborhood $N(v)$ of a vertex $v \in V$ is ordered clockwise with respect to the embedding of G. Such an order can be computed in polynomial-time, by comparing the angle of the vertical bar which crosses $f_V(v)$ with the angle of the straight line which crosses $f_V(v)$ and $f_V(w)$ at the point $f_V(v)$, where $w \in N(v)$. Let $\{v,w\} \in E$ be an edge such that v is at position $i \in \{1,\dots,6\}$ in the neighborhood ordering of w and w at position $j \in \{1,\dots,6\}$ in the neighborhood ordering of v. We introduce an *edge-gadget* $G_{\{v,w\}}$ which is a path consisting of $(\ell+1) \cdot k' - 1$ vertices, where one endpoint is adjacent to each vertex in $C_{(v,j)}$ and the other endpoint is adjacent to each vertex in $C_{(w,i)}$. Hence, a path between two vertex-gadgets has length at least $(\ell+1) \cdot k' + 1$. See Figure 4.6 for an example.

Next, we choose connector sets $C_{(s,i')}$ and $C_{(z,j')}$ such that no vertex $v \in C_{(s,i')} \cup C_{(z,j')}$ is adjacent to a vertex from an edge-gadget. Such i' and j' always exist because the degree of s and z is three. Now, we add two special vertices s' and z' and edges between s' and each vertex in $C_{(s,i')}$, as well as between z' and each vertex in $C_{(z,j')}$. See Figure 4.6 for an example. Finally, we set

$$\ell' := 2 + (\ell+1) \cdot k' + \ell\left((\ell+1) \cdot k' + 1\right).$$

One can observe that G' is a planar graph and

$$|V(G')| = |V| \cdot (k'+1) + |E| \cdot ((\ell+1) \cdot k' - 1) + 2.$$

Hence, G' can be computed in polynomial-time.

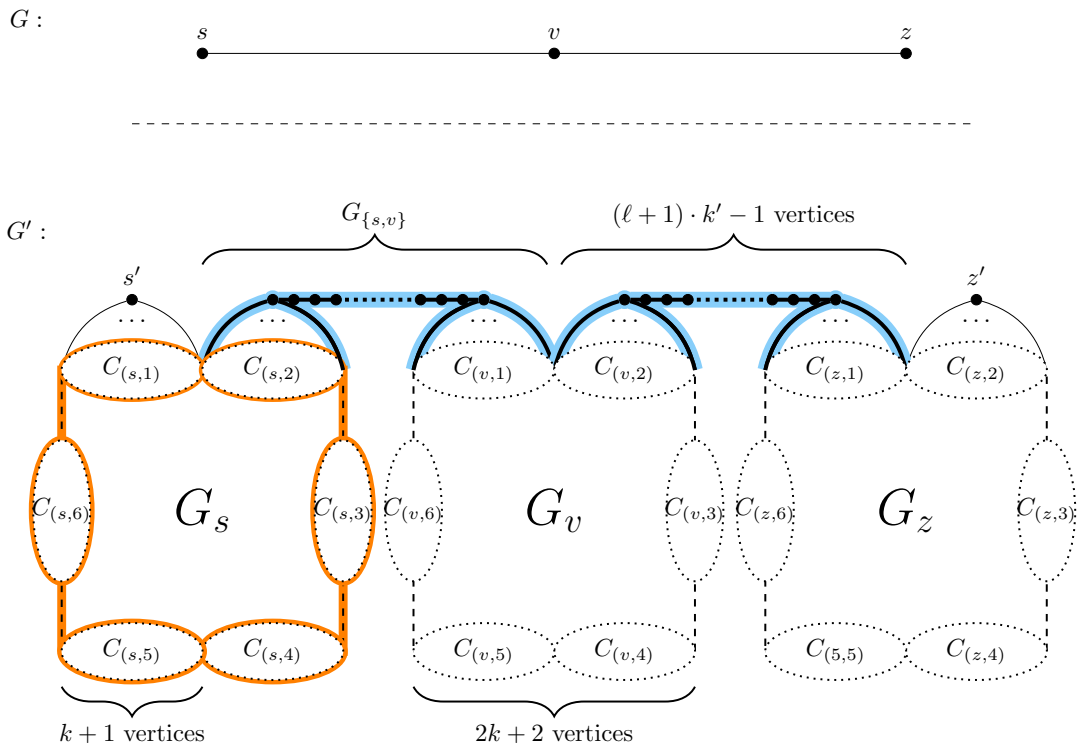


Figure 4.6: A planar graph G (top) and the obtained G' in the reduction from Lemma 4.21. The vertex-gadget G_s of s is orange (dashed/dotted) and the edge-gadgets $G_{\{s,v\}}$ and $G_{\{v,z\}}$ are blue (solid/dotted).

Correctness. We claim that I is a yes-instance if and only if O is a yes-instance.

$\Rightarrow$: Let I be a yes-instance. Thus, there is a solution $C \subset E$ of size at most k such that there is no (s,z)-path of length at most ℓ in $G \setminus C$. We construct a set $S \subset V(G')$ of size $|C| \leqslant k$ by taking for each $\{v,w\} \in C$ one arbitrary vertex from the edge-gadget $G_{\{v,w\}}$ into S.

Assume towards a contradiction that there is an (s',z')-path P' of length at most ℓ' in $G' - S$. Without loss of generality, we assume that P' is cycle free. Since a path between two vertex-gadgets has length at least $(\ell+1)\cdot k' + 1$, we know that P' goes through at most ℓ edge-gadgets. Otherwise P' would be of length at least

$$\begin{aligned} &\overbrace{\begin{array}{c}\text{edge from } s' \text{ to } G_s \\ \text{and from } G_z \text{ to } z'\end{array}}^{2} + (\ell+1)\cdot\big[(\ell+1)\cdot k' + 1\big] \\ =\ &\underbrace{2 + (\ell+1)\cdot k' + \ell\cdot\big[(\ell+1)\cdot k' + 1\big]}_{=\ell'} + 1. \end{aligned}$$

Now, we reconstruct the corresponding (s,z)-path P in G to P' by taking an edge $e \in E$ into P if P' goes through the edge-gadget G_e. Hence, the length of P is at most ℓ. There is no (s,z)-path of length ℓ in $G \setminus C$, and therefore there exists an edge $e \in C$ which is

in P. This contradicts P' being an (s', z')-path in $G - S$ as due to the construction of S, we know that P' visits a vertex in S. Consequently, there is no (s', z')-path of length at most ℓ' in $G' - S$ and O is a yes-instance.

$\Leftarrow$: Let O be a yes-instance. Thus, there is a solution $S \subseteq V(G')$ of size at most k such that there is no (s', z')-path of length at most ℓ' in $G' - S$. We construct an edge set $C \subseteq E$ of size at most k by taking $\{v, w\} \in E$ into C if there is a $y \in V(G_{\{v,w\}}) \cap S$.

Assume towards a contradiction that there is a cycle-free (s, z)-path of length ℓ in $G \setminus C$. We reconstruct an $(s'z')$-path P' in G' which corresponds to P as follows. First, we take an edge $\{s', v\} \in E(G')$ such that $v \in C_{(s,i')} \setminus S$. Such a v always exists, because $|C_{(s,i')}| = k + 1$ and $|S| \leqslant k$. Let $\{s, w\} \in E$ be the first edge of P. and w at position i in the neighborhood ordering of s. Then we add a (v, v')-path P_s in $G_s - S$, such that $v' \in C_{(s,i)}$ S. We claim that such a (v, v')-path P_s always exists in $G_s - S$.

Claim 4.22. *Let G_v be a vertex-gadget and $C_{(v,i)}, C_{(v,j)}$ two connector sets of G_v, where $i, j \in \{1, \ldots, 6\}$ with $i \neq j$. Then, for each vertex set $S \subseteq V(G_v)$ of size at most k it holds that there is a (v_1, v_2)-path of length at most $(2k + 2)^2 - 1$, where $v_1 \in C_{(v,i)} \setminus S$ and $v_2 \in C_{(v,j)} \setminus S$.*

For now, we assume that this claim is true and prove it directly after this proof.

Now, we take an edge-gadget G_e into P' if e is in P. Recall, that an edge-gadget is a path of length $(\ell+1)\cdot k'+1$. One can observe that, because of Claim 4.22, we can connect the edge-gadgets $G_{\{v_1,v_2\}}, G_{\{v_2,v_3\}}$ of two consecutive edges $\{v_1, v_2\}, \{v_2, v_3\}$ in P by a path of length at most k' in G_{v_2}. Let $\{v_\ell, z\}$ be the last edge in P, v_ℓ be at position j in the neighborhood ordering of z, $v \in C_{(z,j)}$, and $v' \in C_{z,j'}$. We add a (v, v')-path of length k' in $G_z - S$ by Claim 4.22. Note that P' visits at most $\ell + 1$ vertex-gadgets and ℓ edge-gadgets. The length of P' is at most

$$2 + (\ell + 1) \cdot k' + \ell \left[(\ell + 1) \cdot k' + 1\right] = \ell.$$

Hence, there is a vertex $x \in S$ which is visited by P'. The vertex x must be part of an edge-gadget $G_{e'}$, because by the construction of P', the strict (s', z')-path P' does not visit any vertex of a vertex-gadget which is also in S. Thus, $e' \in C$. This contradicts P being a strict (s, z)-path in $G \setminus C$. Consequently, there is no (s, z)-path of length at most ℓ in $G - C$ and I is a yes-instance. □

Proof of Claim 4.22. Let G_v be a vertex-gadget and $C_{(v,i)}, C_{(v,j)}$ two connector sets of G_v, where $i, j \in \{1, \ldots, 6\}$ and $i \neq j$. We add vertices s and z and edges $\{s, s'\}$ and $\{z, z'\}$ to G_v, where $s' \in C_{(v,i)}$ and $z' \in C_{(v,j)}$. There are $\binom{6}{2}$ different cases in which $i \neq j$. In all of them it is obvious that there are $k + 1$ vertex-disjoint (s, z)-paths. Consider Figure 4.7 for examples.

By Menger's Theorem (Theorem 2.5), we know that a (s, z)-separator is at least of size $k+1$. Hence, for any vertex set $S \subseteq V(G_v) \setminus \{s, z\}$ of size at most k, there is at least one (s, z)-path in G_v which visits one vertex from $C_{(v,i)}$ and one vertex from $C_{(v,j)}$. □

Note that Lemma 4.21 settles an open question by Fluschnik et al. [Flu+16].

One can observe that in the reduction of Theorem 3.1 from Length-Bounded (s, z)-Separation to Strict (s, z)-Separation the underlying graph of the temporal graph

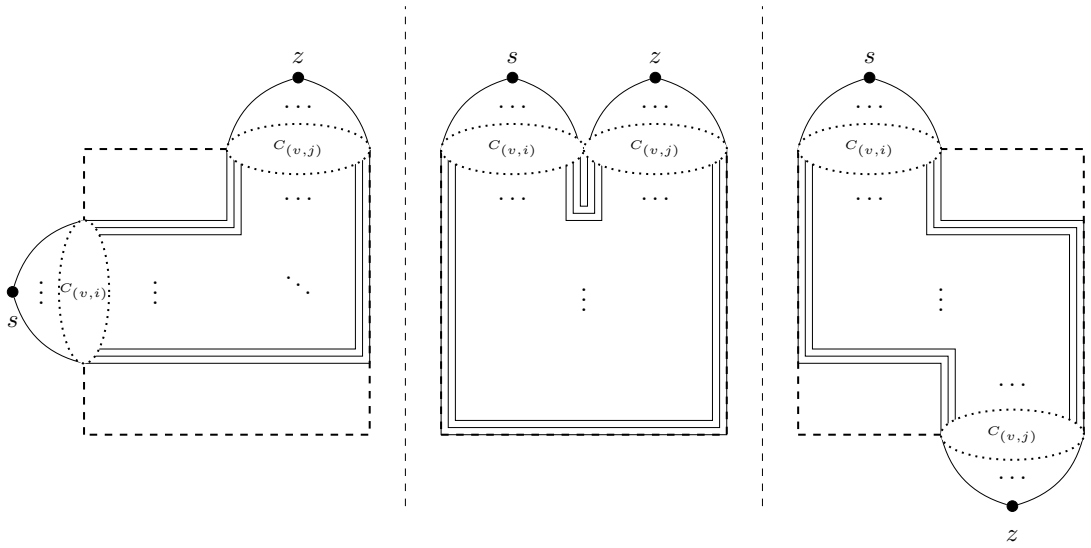


Figure 4.7: Three cases from the proof of Claim 4.22. The grid is pictured by a dashed square, the connector sets are pictured by dotted ellipsoids and the vertex disjoint paths are pictured by lines. The rest of the cases are analogous to these three cases.

from Strict (s, z)-Separation is equal to the input graph from Length-Bounded (s, z)-Separation, and therefore this reduction preserves the property of planarity. Furthermore, we can observe that the reduction of Corollary 3.2 from Strict (s, z)-Separation to Non-Strict (s, z)-Separation also preserves the property of planarity of the temporal graph. Consequently, we have proven the following theorem.

Theorem 4.23. (Non-)Strict (s, z)-Separation *is* NP*-hard on temporal planar graphs.*

From Theorem 3.4, we know that (Non-)Strict (s, z)-Separation is not fixed-parameter tractable when parameterized by the maximum label τ, unless P = NP. Nevertheless, if we restrict the class of underlying graphs, then we can use the algorithm from Theorem 4.6 to show fixed-parameter tractability of Strict (s, z)-Separation when parameterized by the maximum label τ.

Definition 4.24. The *local treewidth* of a graph G is the function $\mathrm{ltw}^G : \mathbb{N} \to \mathbb{N}$ defined as

$$\mathrm{ltw}^G(r) := \max\{\mathrm{tw}(G[N_r^G[v]]) \mid v \in V(G)\}.$$

A class $\mathcal{C}$ of graphs has *bounded local treewidth*, if there is a computable function $h : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{ltw}^G(r) \leqslant h(r)$ for all graphs in $\mathcal{C}$ and $r \in \mathbb{N}$.

For example the class of planar graphs has bounded local treewidth with the function $h : \mathbb{N} \to \mathbb{N}, r \mapsto 3r$ [FG06]. We refer to Flum and Grohe [FG06] for more details.

Corollary 4.25. Strict (s, z)-Separation *on the class of temporal graphs where the underlying graph has bounded local treewidth is fixed-parameter tractable when parameterized by the maximum label* τ*.*

Proof. Let $I = (G = (V, E, \tau), s, z, k)$ be a Strict (s, z)-Separation input instance, where the underlying graph $G_\downarrow$ has bounded local treewidth. Hence, there is a computable function $h : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{tw}(G_\downarrow[N_r^{G_\downarrow}[v]]) \leqslant h(r)$. Let $V' = N_\tau^{G_\downarrow}[s]$. We claim that P is a strict (s, z)-path in G if and only if P is a strict (s, z)-path in $G[V']$.

$\Rightarrow$: Let P be a strict (s, z)-path in G. Since for all consecutive time-edges $(e_1, t_1), (e_2, t_2)$ in P, label t_1 is smaller than t_2, we know that P is of length at most τ. For each vertex $v \in V(P)$ there is a path from s to v of length at most τ in $G_\downarrow$ and therefore $v \in N_\tau^{G_\downarrow}[s] = V'$. Hence, P exists in $G[V']$.

$\Leftarrow$: This direction is easy to see because $G[V']$ is a temporal subgraph of G and therefore every strict (s, z)-path in $G[V']$ is also a strict (s, z)-path in G.

Thus, it is sufficient to solve Strict (s, z)-Separation on $G[V']$. The treewidth of the underlying graph $G[V']_\downarrow$ of $G[V']$ is upper-bounded by $h(\tau)$ By Theorem 4.6, we can solve the instance I in $\mathcal{O}((2\tau + 2)^{\mathcal{O}((h(\tau)+2\cdot\binom{h(\tau)}{2})\cdot\tau)^3)} \cdot h(\tau) + 2 \cdot \binom{h(\tau)}{2} \cdot \tau \cdot |V| \cdot |E| + |E|^2)$ time. □

As already mentioned above, for every planar graph G, $\mathrm{ltw}^G(r) \leqslant 3r$ [FG06]. Hence, from Corollary 4.25, we can derivei the following fixed-parameter tractability result.

Corollary 4.26. Strict (s, z)-Separation *on temporal planar graphs is fixed-parameter tractable when parameterized by the maximum label* τ.

Unfortunately, the idea of Corollary 4.25 is not transferable to Non-Strict (s, z)-Separation. It can be seen as further research opportunity to investigate the computational complexity of Non-Strict (s, z)-Separation on temporal planar graphs, when parameterized by the maximum label τ.

4.3 No Polynomial Kernels

We already discovered that Strict (s, z)-Separation is in FPT when parameterized by $k+\tau$, $\tau+\mathrm{tw}_\downarrow$, and $|V|$, and Non-Strict (s, z)-Separation is in FPT when parameterized by $k+\tau+\mathrm{tw}_{\max}$, $\tau+\mathrm{tw}_\downarrow$, and $|V|$, where k is the solution size, τ is the maximum label, $\mathrm{tw}_\downarrow$ is the underlying treewidth, $\mathrm{tw}_{\max}$ is the layer treewidth, and $|V|$ is the number of vertices. This implies the existence of kernels for Strict (s, z)-Separation and Non-Strict (s, z)-Separation with these parameterizations [DF13, Proposition 4.8.1]. But all of them are of exponential size with respect to the parameterization.

This section is devoted to showing lower bounds on the size of these kernels. We show that, unless NP $\subseteq$ coNP/poly, there is no polynomial kernel for (Non-)Strict (s, z)-Separation when parameterized by $k+\tau+\mathrm{tw}_\downarrow+\Delta$, where k is the solution size, τ is the maximum label, $\mathrm{tw}_\downarrow$ is the underlying treewidth, and Δ is the maximum degree in the temporal graph. Note that $\mathrm{tw}_{\max} \leqslant \mathrm{tw}_\downarrow$, where $\mathrm{tw}_\downarrow$ is underlying treewidth and $\mathrm{tw}_{\max}$ is the layer treewidth.

On the one hand, this is a negative result. If we consider instances of a problem P with small parameter k, then we would like to have a small kernel size because in this case we can shrink an arbitrarily large instance to a small kernel in polynomial-time.

One the other hand, this is a positive result. Suppose we are a vendor of a general purpose solver for the problem P. Since our customers want to solve all kinds of instances of the problem P, we probably cannot identify a specific parameter to be small in general. Hence, in the first place fixed-parameter algorithms are not especially attractive to us. But in contrast to that, regardless of the parameterization for which they are useful, reduction rules are fast and help us to shrink the size of the input instance. A *hard instance* is an input instance, onto which no reduction rule can applied to. Now, we have to solve hard instances of size n. Let r be a parameter such that (P, r) admits a kernel of size $f(r)$. Hence, we can upper bound the size of any hard instance by $f(r)$, and consequently, we also have the lower bound $r \geqslant f^{-1}(n)$ on the parameter. For example, if f is linear then $r \in \mathcal{O}(n)$ and hence, the benefit of a fixed-parameter algorithms for the parameter r is always highly limited. But if we have that f is exponential in r, then the lower bound on r for a hard instance is logarithmic in r and thus, r can be rather small with respect to the input size n.

Consequently, we have a high interest in developing fast fixed-parameter algorithms for parameters whose values are easy to determine. In such a scenario, we can enhance our general purpose solver such that it checks whether one of these parameters is small in a hard instance and apply the fixed-parameter algorithm.

One can observe that the parameters k, τ, $|V|$, and Δ can be computed in polynomial-time, where k is the solution size, τ is the maximum label, $|V|$ is the number of vertices, and Δ is the maximum degree in the temporal graph.

Bodlaender et al. [Bod+09] developed a framework which can be helpful to show a lower bound on a kernel size. Note that this framework assumes $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$ which is widely believed, because otherwise the polynomial hierarchy [Sto76] collapses as Fortnow and Santhanam [FS11] have shown. Here, we will use a generalized version of this framework which is also used by Fluschnik et al. [Flu+16].

Definition 4.27 (Polynomial Equivalence Relation)**.** Given an NP-hard problem L, an equivalence relation $\mathcal{R}$ on the instance of L is a *polynomial equivalence relation* if

(i) one can decide for any two instances in time polynomial in their sizes whether they belong to the same equivalence class, and

(ii) for any finite set S of instances, R partitions the set into at most $(\max_{x \in S} |x|)^{O(1)}$ equivalence classes.

Definition 4.28 (OR-Cross-Composition)**.** Given an NP-complete problem L, a parameterized problem P, and a polynomial equivalence relation $\mathcal{R}$ on the instances of L, an OR-cross-composition of L into P (with respect to $\mathcal{R}$) is an algorithm that takes p $\mathcal{R}$-equivalent instances $I_1, \ldots, I_p$ of L and constructs in time polynomial in $\sum_{i=1}^{p} |I_i|$ an instance (I, k) of P such that

(i) k is polynomially upper-bounded in $\max_{i \in \{1,\ldots,p\}} |I_i| + \log(p)$ and

(ii) (I, k) is a yes-instance of P if and only if there is at least one $i \in \{1, \ldots, p\}$ such that I_i is a yes-instance of L.

If a parameterized problem P admits an OR-cross-composition for some NP-complete problem L, then P does not admit a polynomial kernel with respect to its parameterization, unless NP $\subseteq$ coNP/poly [BJK14]. Note that we can assume without loss of generality that $p = 2^q$, because we can add trivial no-instances to reach a power of two for some $q \in \mathbb{N}$.

Fluschnik et al. [Flu+16] showed that, unless NP $\subseteq$ coNP/poly, Length-Bounded (s, z)-Cut does not admit a polynomial kernel, when parameterized by the solution size and the maximum path length. Unfortunately, they did not prove that for Length-Bounded (s, z)-Separation, but conjectured that the same result can also be shown for this problem. This conjecture seems to be highly convincing, since they even state how to adjust their construction. They achieved the no polynomial kernel result for Length-Bounded (s, z)-Cut by introducing the so-called *T-fractal*. Later, we extend this idea by constructing a temporal T-fractal.

Definition 4.29. For $q \geqslant 1$, the *T-fractal* Δ_q is the graph constructed as follows:

(i) Set the graph $\Delta_0 := (\{\sigma, \delta\}, \{\{\sigma, \delta\}\})$ with $\{\sigma, \delta\}$ being a *marked edge* with endpoints σ and δ, subsequently referred to as special vertices.

(ii) Let F be the set of marked edges and set $F' = \emptyset$.

(iii) For each edge $e \in F$, add a new vertex and connect it by two new edges with the endpoints of e, and add the two added edges into F'.

(iv) Unmark all edges in F and mark all edges in F'.

(v) Repeat (ii)-(iv) $q - 1$ times.

Roughly speaking, a T-fractal can be constructed by iteratively putting triangles on top of each other.

Figure 4.8 gives an idea how Fluschnik et al. [Flu+16] have used T-fractal to OR-cross-compose $p = 8$ input instances of Length-Bounded (s, z)-Cut, all with the same parameters, to one output instance of Length-Bounded (s, z)-Cut where σ and δ are the special vertices. One can observe that each possible (σ, δ)-cut selects one input instance such that every (σ, δ)-path is going through this input instance. For more details, we refer to Fluschnik et al. [Flu+16].

Lemma 4.30. *Unless* NP $\subseteq$ coNP/poly, Non-Strict (s, z)-Separation *does not admit a polynomial kernel when parameterized by* $k + \tau + \mathrm{tw}_\downarrow$, *where*

- k *is the solution size,*
- τ *is the maximum label, and*
- $\mathrm{tw}_\downarrow$ *is the treewidth of the underlying graph.*

Proof. We OR-cross-compose $p = 2^q$ instances of the Non-Strict (s, z)-Separation problem into one instance of the Non-Strict (s, z)-Separation problem with respect to $k + \tau$ and afterwards construct a tree-decomposition for the underlying graph whose width is also an upper bound for the treewidth of the layer treewidth.

A Non-Strict (s, z)-Separation instance $(G_i = (V_i, E_i, \tau_i), s_i, z_i, k_i)$ is called *bad* if $\max\{k_i, \tau_i\} > |E_i|$ or $\min\{k_i, \tau_i\} < 1$. We define the polynomial equivalence relation $\mathcal{R}$ on the instances of the Non-Strict (s, z)-Separation problem as follows: two

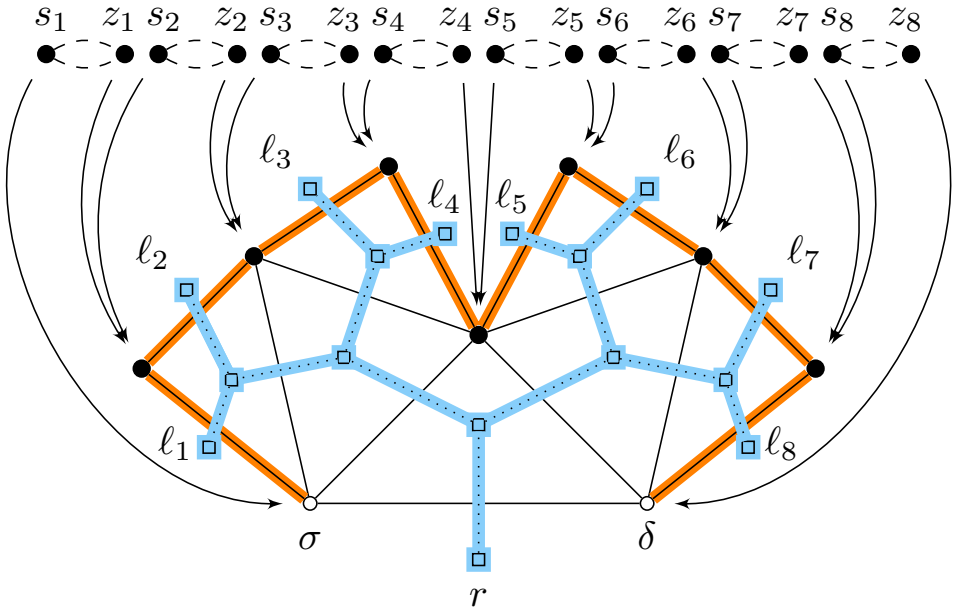


Figure 4.8: A visualization of the OR-cross-composition of [Flu+16]. Eight instances of Length-Bounded (s, z)-Cut (top) with special vertices s_i, z_i, where $i \in \{1, \ldots, 8\}$. A T-fractal Δ_3 (bottom) with special vertices σ and δ (white vertices). The (σ, δ)-path on the outer boundary (solid edges with orange background). A tree (rectangle vertices, dotted edge with blue background) which is not part of the OR-cross-composition but shows all possible (σ, δ)-cuts in Δ_3 of size 4. Every (r, ℓ_i)-path symbolize a (σ, δ)-cut in Δ_3 of size 4, where $i \in \{1, \ldots, 8\}$. The arrows show which special vertex of a Length-Bounded (s, z)-Cut instance is merged with which vertex of Δ_3.

instances $(G_i = (V_i, E_i, \tau_i), s_i, z_i, k_i)$ and $(G_j = (V_j, E_j, \tau_j), s_j, z_j, k_j)$ are $\mathcal{R}$-equivalent if and only if $k_i = k_j$ and $\tau_i = \tau_j$, or both are bad. Clearly, the relation $\mathcal{R}$ fulfills condition (i) of Definition 4.27. Observe that the number of equivalence classes of a finite set $J := \{I_1, \ldots, I_p\}$ is upper-bounded by ${m_j}^2 + 1$, where m_j is the maximum size of a time-edge set over the instances in J, hence condition (ii) of Definition 4.27 holds as well.

Thus, we consider $p = 2^q$ $\mathcal{R}$-equivalent input instances $I_i = (G_i = (V_i, E_i, \tau), s_i, z_i, k)$ and OR-cross-compose into the output instance $\hat{O} := (G = (V, E, \tau), s, z, k')$, where $i \in \{1, \ldots, p\}$ and

$$k' := (q + 1) \cdot (k + 1) + k = (\log(p) + 1) \cdot (k + 1) + k.$$

One can already see that the parameter $k' + \tau$ of $\hat{O}$ is polynomially upper-bounded in $\max_{i \in \{1,\ldots,p\}} |I_i| + \log(p)$.

We construct a *temporal T-fractal* $\hat{\triangle}_q := (V_q, E_q, \tau)$, which is defined analogously to the T-fractal $\triangle_q$ from Definition 4.29. For each $v \in V(\triangle_q)$ there is a set of $k' + 1$ vertices which is called the *node-vertex set* of v. For each edge $\{v, w\} \in E(\triangle_q)$ we add a set of $k + 1$ vertices and connect each of them with all vertices in the node-vertex sets

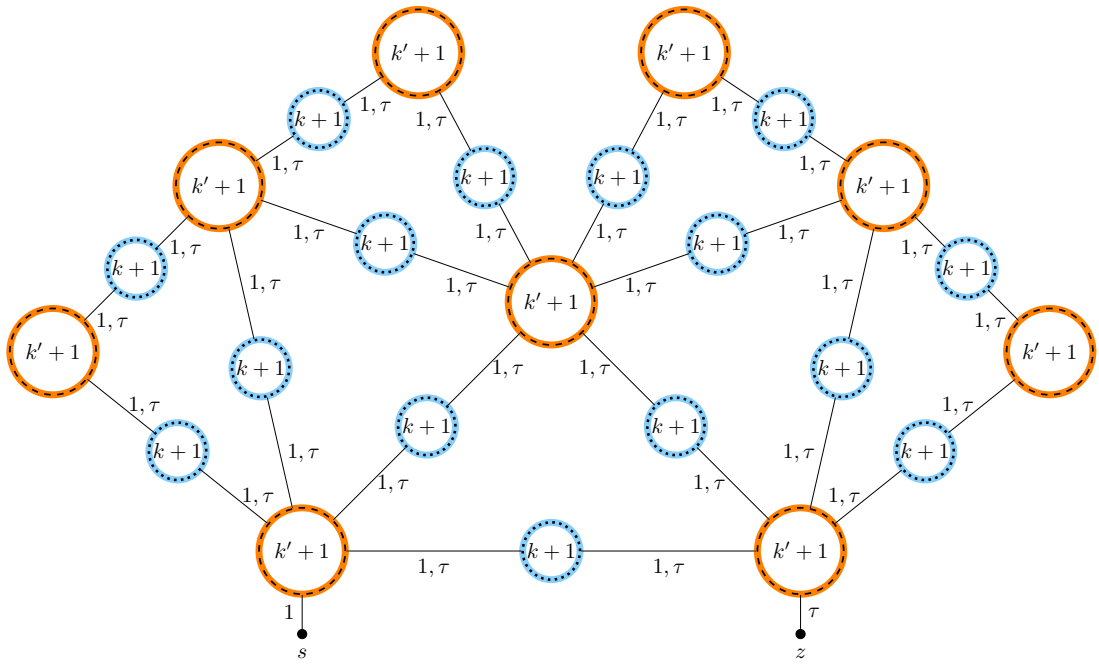


Figure 4.9: The temporal T-fractal $\hat{\Delta}_3$. The node-vertex sets are highlighted by dashed circles (orange). The edge-vertex sets are highlighted by dotted circles (blue). A time-edge between two sets of vertices A and B means that for all vertices $a \in A$ and $b \in B$ there is such a time-edge between a and b.

of v and w with two time-edges labeled by one and τ, respectively. We will call this set of vertices the *edge-vertex set* of $\{v, w\}$. Finally, we add the special vertices s and z such that there is a time-edge from s to all vertices in the node-vertex set of σ with label one and from z to all vertices in the node-vertex set of δ with label τ. See Figure 4.9 for an example.

As in the construction from Fluschnik et al. [Flu+16] we put the input instances on the outer boundary of $\hat{\triangle}_q$. Let v_i be the i-th internal vertex of a path from σ to δ on the outer boundary of $\triangle_q$ and let C_i be the corresponding node-vertex set of v_i in $\hat{\triangle}_q$. For clarification, σ is v_1 and δ is v_{p+1}.

Fluschnik et al. [Flu+16] stated that $\triangle_q$ has $p+1$ vertices and $2^{q+1}-1 = 2p-1$ edges. Thus, we know the number of vertices in $\hat{\triangle}_q$ is

$$|V(\hat{\triangle}_q)| = 2 + (p+1) \cdot (k'+1).$$

Each vertex from an edge-vertex sets has two time-edges to $2(k'+1)$ vertices plus $2(k'+1)$ time-edges for the special vertices s and z. Therefore, we have that

$$|E(\hat{\triangle}_q)| = (2p-1) \cdot (k+1) \cdot 4(k'+1) + 2(k'+1).$$

Now, we create G by taking $\hat{\triangle}_q$ and

- add each $G_i - \{s_i, z_i\}$ to G,

- add for each time-edge $(\{s_i, w\}, t) \in E(G_i)$ a time-edge $(\{v, w\}, t) \in E(G)$ for all vertices v from the node-vertex set of v_i, and
- add for each time-edge $(\{w, z_i\}, t) \in E(G_i)$ a time-edge $(\{v, w\}, t) \in E(G)$ for all vertices v from the node-vertex set of v_{i+1}.

One can observe that the time required to compute G can be upper-bounded by a polynomial in $\sum_{i=1}^{p} |I_i|$.

Note that each node-vertex set has more than k' vertices and there is no non-strict (s, z)-separator of size k' which can contain all vertices of a node-vertex set. Hence, only vertices from edge-vertex sets are relevant in a non-strict (s, z)-separator. To be precise, let $S = X \uplus Y$ be a non-strict (s, z)-separator in $\hat{\triangle}_q$, where X only contains vertices from node-vertex sets and Y only contains vertices from edge-vertex sets. Then, Y is also a non-strict (s, z)-separator in $\hat{\triangle}_q$.

We know that the T-fractal $\triangle_q$ has a minimum cut size of exactly $q + 1$ and that there are p different cuts of size $q+1$ [Flu+16]. Analogously, we can conclude that a non-strict (s, z)-separator of $\hat{\triangle}_q$ must contain $q + 1$ edge-vertex sets. Note that k' is chosen such that a non-strict (s, z)-separator of size at most k' cannot contain $q+2$ edge-vertex sets. Hence, we know that $\hat{\triangle}_q$ will work as an instance selector in G by selecting $q + 1$ edge-vertex sets. We refer to Fluschnik et al. [Flu+16] for more information about the instance selection.

For all $i \in \{1, \ldots, p\}$ and each non-strict (s_i, z_i)-path P in G_i there is a non-strict (s, z)-path in G, because of the construction of $\hat{\triangle}_q$ there is a non-strict (s, s_i)-path P_- in G with departure and arrival time one, where $s_i \in C_i$. Furthermore, there is non-strict (z_i, z)-path P_+ in G with departure and arrival time τ, where $z_i \in C_{i+1}$. The concatenation of P_-,P, and P_+ gives a non-strict (s, z)-path in G.

We claim that there is an $i \in \{1, \ldots, p\}$ such that I_i is a yes-instance if and only if $\hat{O}$ is a yes-instance.

$\Rightarrow$: Assume that there is an $i \in \{1, \ldots, p\}$ such that I_i is a yes-instance. Let $s_i \in C_i$ and $z_i \in C_{i+1}$. Then we can select I_i by taking $q + 1$ edge-vertex sets into S such that there is no non-strict (s, z)-path, no non-strict (s, z_i)-path, and no non-strict (s_i, z)-path in $\hat{\triangle}_q$ which means that all non-strict (s, z)-paths in $G - S$ go through G_i. Note that $|S| = (q+1)(k+1)$. Since I_i is a yes-instance, we know there is a non-strict (s_i, z_i)-separator S_i of size at most k in G_i. Hence, $S \cup S_i$ is a non-strict (s, z)-separator of size k'.

$\Leftarrow$: Now let each input instance be a no-instance. Assume towards a contradiction there is a non-strict (s, z)-separator S in G of size at most k'. Since $\hat{\triangle}_q$ is a subgraph of G which contains s and z, S is also a non-strict (s, z)-separator in $\hat{\triangle}_q$. Let $S = S_\triangle \uplus S'$, where $S_\triangle$ is the non-strict (s, z)-separator of $\triangle_q$. We need $q+1$ edge-vertex sets in $S_\triangle$ otherwise $S_\triangle$ cannot be a non-strict (s, z)-separator in $\hat{\triangle}_q$. Note that the number of vertices of $q + 2$ edge-vertex sets is more than k'. There are p different (σ, δ)-cuts in $\triangle_q$ and for each of them there is a v_i in the connected component of σ where the v_{i+1} is in the connected component of δ [Flu+16]. Thus, there are $s_i \in C_i$ and $z_i \in C_{i+1}$ such that there is a non-strict (s, s_i)-path and a non-strict (z_i, z)-path in $G - S$. Note that $|S \setminus S_\triangle| \leqslant k$. There is no non-strict (s_i, z_i)-separator of size at most k in G_i. Hence, $|S| > k'$. This contradicts $\hat{O}$ being a no-instance.

Next, we are going to construct a tree-decomposition for the underlying graph of G. Baker [Bak94] showed that outerplanar graphs have treewidth two. Since the T-fractal Δ_q is outerplanar, it has treewidth at most two. Let $\mathcal{T} = (T, (B_x)_{x \in V(T)})$ be such a tree-decomposition of Δ_q of width at most two. Now, we construct a tree-decomposition $\mathcal{T}' = (T', (B'_x)_{x \in V(T')})$ for the underlying graph of $\hat{\Delta}_q - \{s, z\}$, by

(i) copying $\mathcal{T}$ into $\mathcal{T}'$,

(ii) replacing each vertex $v \in B'_x$ with its node-vertex set, and

(iii) if $v, w \in B_x$, where $\{v, w\}$ is an edge of Δ_q, then we introduce the edge-vertex set of $\{v, w\}$ intro B'_x.

The width of $\mathcal{T}'$ is at most $3 \cdot (k' + 1) + 3 \cdot (k + 1) = 3 \cdot (k' + k + 2)$, because each bag of $\mathcal{T}$ contains at most three vertices and edges. We extend $\mathcal{T}'$ such that all vertices of $G_i - \{s_i, z_i\}$ are in B'_x if v_i or v_{i+1} is in B_x, where $i \in \{1, \ldots, p\}$. Note that this is a tree-decomposition of $G - \{s, z\}$. Finally, we add s and z to every bag of $\mathcal{T}'$ and obtain a tree-decomposition of the underlying graph of G. Since every bag of $\mathcal{T}$ has at most three vertices and every input instance is connected to two node-vertex sets in $\hat{\triangle}_q$, we can upper bound the width of $\mathcal{T}'$ by $3 \cdot (k' + k + 2) + 6 \cdot n_{\max}$, where $n_{\max} = \max_{i \in \{1, \ldots, p\}} |V_i|$. Hence, the treewidth of the underlying graph of G is upper bounded by a polynomial in $\max_{i \in \{1, \ldots, p\}} |I_i| + \log(p)$. □

We can adjust the temporal T-fractal further to extend the result from Lemma 4.30 to Strict (s, z)-Separation.

Lemma 4.31. *Unless* NP ⊆ coNP/poly, Strict (s, z)-Separation *does not admit a polynomial kernel, when parameterized by* $k + \tau + \mathrm{tw}_\downarrow$, *where*

- k *is the solution size,*
- τ *is the maximum label, and*
- $\mathrm{tw}_\downarrow$ *is the treewidth of the underlying graph.*

Proof. We will OR-cross-compose from $p = 2^q$ input instances $I_1, \ldots, I_p$ of Strict (s, z)-Separation to one output instance $\hat{O} := (G = (V, E, \tau'), s, z, k')$ of Strict (s, z)-Separation with respect to $k + \tau + \mathrm{tw}_\downarrow$ and use the polynomial equivalence relation $\mathcal{R}$ from Lemma 4.30 again, but this time for Strict (s, z)-Separation. An input instance is denoted by $I_i = (G_i = (V_i, E_i, \tau), s_i, z_i, k)$. The OR-cross-composition works essentially in the same way as in Lemma 4.30. There are three adjustments.

First, we have to ensure that each input instance is reachable from s. Let $D \subseteq E(\triangle_q)$. Fluschnik et al. [Flu+16] showed that every vertex in the connected component of σ in $\triangle_q \setminus D$ can be reached with a path of length at most $q + |D| + 1$ from σ. Let $s_i \in C_i$, where $i \in \{1, \ldots, p\}$ and C_i be the i-th node-vertex set on the outer boundary of $\hat{\triangle}_q$. This implies that after the removal of $q+1$ edge-vertex sets from $\hat{\triangle}_q$, either there is a non-strict (s, s_i)-path with departure and arrival time one of length at most $\ell := 2(2q + 2)$, or there is no non-strict (s, s_i)-path. Now we start to construct the *strict temporal T-fractal* $\overline{\triangle}_q$ by taking all vertices from $\hat{\triangle}_q$ and add for each time-edge $(\{v, w\}, 1) \in E(\hat{\triangle}_q)$ exactly ℓ many time-edges $(\{v, w\}, j) \in E(\overline{\triangle}_q)$, where $j \in \{1, \ldots, \ell\}$. Hence, for each

non-strict (s, s_i)-path in $\hat{\triangle}_q$ with departure and arrival time one there is a strict (s, s_i)-path with departure time one and arrival time at most ℓ.

Second, we have to shift each label in an input instance by ℓ time steps. Therefore, each time-edge $(\{v, w\}, t)$ in G_i will be replaced by a time-edge $(\{v, w\}, t+\ell)$. The shifted instance of G_i is denoted by $\vec{G}_i$. One can observe that for each strict (s_i, z_i)-path with departure time $a \geqslant 1$ and arrival time $b \leqslant \tau$ in G_i there is a strict (s_i, z_i)-path with departure time $\ell + a$ and arrival time $\ell + b$ in $\vec{G}_i$.

Third, analogously to the first part we have to ensure that we can reach z from each input instance. Let $z_i \in C_i$, where $i \in \{2, \ldots, p+1\}$ and C_i be the i-th node-vertex set on the outer boundary of $\hat{\triangle}_q$. For each time-edge $(\{v, w\}, \tau) \in E(\hat{\triangle}_q)$ we add ℓ many time-edges $(\{v, w\}, \ell + \tau + j) \in E(\overline{\triangle}_q)$, where $j \in \{1, \ldots, \ell\}$. Thus, for each non-strict (z_i, z)-path with departure and arrival time τ in $\hat{\triangle}_q$ there is a strict (z_i, z)-path with departure time $\tau + \ell + 1$ and arrival time at most $\tau + 2\ell$.

One can observe that $|V(\overline{\triangle}_q)| = |V(\hat{\triangle}_q)|$ and that $|E(\overline{\triangle}_q)| = |E(\hat{\triangle}_q)| \cdot \ell$.

Now we OR-cross-compose from the instances $\vec{I}_1, \ldots, \vec{I}_p$ into $\hat{O}$ in exactly the same way as we did in Lemma 4.30 but instead of $\hat{\triangle}_q$ the T-fractal $\overline{\triangle}_q$ is used, where $\vec{I}_i := (\vec{G}_i, s_i, z_i, k)$ for all $i \in \{1, \ldots, p\}$. The first part of the parameter of $\hat{O}$ is

$$\begin{aligned} & k' + \tau' \\ \Leftrightarrow\ & (\log(p) + 1) \cdot (k+1) + k + \tau + 2 \cdot \ell \\ \Leftrightarrow\ & (\log(p) + 1) \cdot (k+1) + k + \tau + 2 \cdot [2(2q+2)] \\ \Leftrightarrow\ & (\log(p) + 1) \cdot (k+1) + k + \tau + 8 \cdot \log(p) + 8. \end{aligned}$$

Furthermore, the underlying graph of $\overline{\triangle}_q$ is identical to the underlying graph of $\hat{\triangle}_q$. Hence, $k' + \tau' + \mathrm{tw}_\downarrow$ is polynomially upper bounded in $\max_{i \in \{1,\ldots,p\}} |I_i| + \log(p)$.

Now, we claim that there is an $i \in \{1, \ldots, p\}$ such that I_i is a yes-instance if and only if $\hat{O}$ is a yes-instance. The correctness argumentation is identical to argumentation in the proof of Lemma 4.30 if we substitute non-strict with strict and adjust the departure and arrival time of P_-, P, and P_+. □

Fluschnik et al. [Flu+16] showed that the maximum degree Δ of a $\triangle_q$ is exactly $2q$. Each vertex from an edge-vertex set in $\hat{\triangle}_q$ has degree $4 \cdot (k'+1)$. If a vertex v from $\triangle_q$ has degree d, then a vertex in the node-vertex set of v has degree $2d \cdot (k+1)$. Furthermore, in $\overline{\triangle}_q$ we have $\ell = \log(p) + 8$ time-edges for each time-edge in $\hat{\triangle}_q$. Thus, the maximum degree in $\hat{\triangle}_q$ is exactly $4q \cdot (k+1)$ and the maximum degree in $\overline{\triangle}_q$ is exactly $4q \cdot (k+1) \cdot \ell$.

We can use this knowledge to strengthen Lemma 4.30 and Lemma 4.31 and finally state the desired theorem.

Theorem 4.32. *Unless* NP ⊆ coNP/poly, (Non-)Strict (s, z)-Separation *does not admit a polynomial kernel, when parameterized by* $k + \tau + \mathrm{tw}_\downarrow + \Delta$, *where*

- k *is the solution size,*
- τ *is the maximum label,*
- $\mathrm{tw}_\downarrow$ *is the treewidth of the underlying graph, and*
- Δ *is the maximum degree in the temporal graph.*

In Section 4.2, we developed a method to maintain planarity from Length-Bounded (s, z)-Cut to Length-Bounded (s, z)-Separation, when the maximum degree is bounded by introducing grids for vertices (see Lemma 4.21). This idea seems to be also applicable in the OR-cross-composition of Lemmata 4.30 and 4.31. Therefore, we conjecture that Theorem 4.32 also holds for temporal planar graphs. Furthermore, we claim that this is the key tool to settle the open question of Fluschnik et al. [Flu+16] whether Length-Bounded (s, z)-Separation has a polynomial kernel on planar graphs, when parameterized by solution size k, the maximum length ℓ of a path, and the treewidth of the input graph.

(Non-)Strict (s, z)-Separation is in FPT, when parameterized by the number $|V|$ of vertices, but we could not determine whether there is a polynomial kernel for this parameter.

Chapter 5

A General Path Model: β-Bounded Paths

In this chapter, we introduce a new path notion for temporal graphs where the difference between the labels of two consecutive time-edges in a path are bounded.

Suppose a disease is going around and we, here called z, do not want to get ill in the next six weeks because we have to perform at the Olympic Games. There is an inoculation for this disease but we would need to reduce our training for one week if we take the inoculation. This is something we cannot afford in the last six weeks before the Olympic Games. Until the Olympic Games begin, we will stay at a training camp where nobody is allowed to have visitors and nobody else is going to compete in the Olympic Games. So, they are willing to do the inoculation if the pressure is high enough. Of course, our training schedule is extremely tight and therefore we can only persuade at most k athletes to take the inoculation. Furthermore, everything is well premeditated in the training camp. We know who also stays in the training camp and which schedule they follow. Unfortunately, some of the athletes already have the disease.

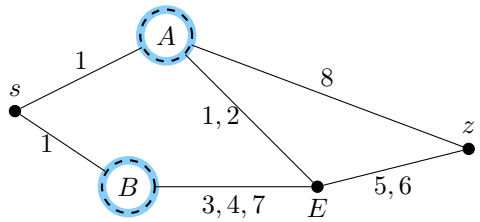
We model a temporal graph for the next weeks as follows: All athletes, including z, are vertices plus a special vertex s. If two athletes meet somewhere, then there is a time-edge between those vertices labeled by the timestamp of the meeting. We replace the vertex v of an athlete who has the disease with vertex set of $k+1$ vertices such that the neighborhood of each new vertex is equal to the neighborhood of v. Then we add a time-edge from s to each new vertex labeled with one. Obviously, a non-strict (s, z)-separator of the temporal graph would tell us which k athletes we can persuade to take the inoculation such that we will not get ill in the next six weeks.

But if there is no non-strict (s, z)-separator, it does not mean that we cannot persuade at most k athletes to take the inoculation such that we will not get ill in the next six weeks, because the immune system will cure the disease after some time β. Hence, if an athlete gets the disease at time point t, then one cannot infect others after time point $t + \beta$. Consider Figure 5.1 for an example.

A $(0, \beta)$-bounded (s, z)-path $P = (e_1, t_1), \ldots, (e_\ell, t_\ell)$ (or *β-bounded (s, z)-path* for short) of length ℓ is a non-strict (s, z)-path of length ℓ where $t_{i+1} - t_i \leqslant \beta$ for all $i \in \{1, \ldots, \ell - 1\}$. In the literature, this model of a path is also known as vertex buffer times [Akr+17] and as waiting time cutoffs [PS11].

Day	Training
(1) Mon	A, E endurance
(2) Tue	A, E max-power
(3) Wed	B, E endurance
(4) Thur	B, E max-power
(5) Fri	E, z endurance
(6) Sat	E, z balance
(7) So	B, E mobilization
(8) Mo	A, z max-power

(a) Schedule



(b) Temporal Graph G

Figure 5.1: Figure 5.1a shows a schedule of a training camp for Alice (A), Bob (B), Emma (E), and us (z). Figure 5.1a shows the temporal graph G obtained by the construction from the introductory example of Chapter 5 for the schedule in Figure 5.1a. We can only persuade at most k athletes to take the inoculation. Alice (A) and Bob (B) just got the disease while Emma (E) is healthy. Hence, Alice and Bob are represented by $k+1$ vertices, respectively (blue/dashed). In this example the disease is contagious for seven days after the infection. Thus, each 6-bounded (s, z)-path in G denotes a way how we can get the disease. Note that we can only get the disease from Emma because on the second Monday (8) Alice is not contagious anymore.

This is a generalization of the non-strict path model because a non-strict (s, z)-path is a τ-bounded (s, z)-path and vice versa. Note that it is not possible to wait at a vertex for an arbitrary amount of time and hence one has to go a cycle to bridge the waiting time. Furthermore, there is a close relation to multi-layer graphs—for each $(0, 0)$-bounded (s, z)-path P_0 in a temporal graph $G = (V, E, \tau)$ there exists an $i \leqslant \tau$ such that there is an (s, z)-path in the layer G_i which visits the same vertices as P_0 and in the same order as P_0.

A *β-bounded (s, z)-separator* is a set $S \subseteq V \setminus \{s, z\}$ of vertices such that there is no β-bounded (s, z)-path in $G - S$.

β-Bounded (s, z)-Separation

Input: A temporal graph $G = (V, E, \tau)$, two distinct vertices s, z, and an integer $k \in \mathbb{N}$.

Question: Is there a β-bounded (s, z)-separator S of size at most k?

In the remaining part of this section, we show that β-Bounded (s, z)-Separation is W[2]-hard when parameterized by the solution size k and that β-Bounded (s, z)-Separation is fixed-parameter tractable when parameterized by the number $|V|$ of vertices. Afterwards, we show that β-Bounded (s, z)-Separation does not admit a polynomial kernel when parameterized by $|V|$, unless NP $\subseteq$ coNP/poly.

Theorem 5.1. *β-Bounded (s, z)-Separation is W[2]-hard when parameterized by the solution size k.*

Proof. We reduce from the Hitting Set problem which is known to be W[2]-hard, when parameterized by the solution size.

Hitting Set Problem
Input: A universe $U = \{u_1, \ldots, u_m\}$, a collection $C = \{C_1, \ldots, C_n\}$ where $C_i \subseteq U$ for all $i \in \{1, \ldots, n\}$ and $k \in \mathbb{N}$
Question: Is there a subset $H \subseteq U$ of size at most k such that $H \cap C_i \neq \emptyset$ for all $i \in \{1, \ldots, n\}$?

Let (U, C, k) be a Hitting Set problem instance, where $|U| = n$ and $|C| = m$. We construct a β-Bounded (s, z)-Separation instance $(G = (V, E, (\beta + 1) \cdot (m - 1) + 1), s, z, k)$ as follows: set $V := U \cup \{s, z\}$. For all $u, u' \in C_i \in C$ add a time-edge $(\{u, u'\}, (i-1) \cdot (\beta + 1) + 1)$ to E, where $u \neq u'$. Fix an arbitrary linear order $f : U \to \{1, \ldots, n\}$ for U. We denote an element $u \in U$ by u_j if and only if $f(u) = j$. For each $C_i \in C$ add time-edges $(\{s, u_\alpha\}, (i-1) \cdot (\beta+1)+1))$ and $(\{u_\beta, z\}, (i-1) \cdot (\beta+1)+1))$ in E, where $\alpha = \min_{u \in C_i} f(u)$ and $\beta = \max_{u \in C_i} f(u)$.

We claim that $S \subseteq (V \setminus \{s, z\})$ is a β-bounded (s, z)-separator of size at most k if and only if S is a solution to the Hitting Set instance (U, C, k).

$\Rightarrow$: Let S be a β-bounded (s, z)-separator in G, $C_i \subseteq C$, and $c_1, \ldots, c_r \in C_i$ be all elements in C_i such that $f(c_j) < f(c_{j+1})$ for all $j \in \{1, \ldots, r-1\}$. There is a β-bounded (s, z)-path $P = (\{s, c_1\}, (i-1) \cdot (\beta + 1) + 1), \ldots, (\{c_r, z\}, (i-1) \cdot (\beta + 1) + 1)$ in G. Hence, S contains at least one vertex of C_i. Thus, S is a solution to the Hitting Set instance (U, C, k).

$\Leftarrow$: Let S be a solution for the Hitting Set instance (U, C, k). Assume towards a contradiction that S is not a β-bounded (s, z)-separator in G. One can observe that $G[P]$ is the layer $(i-1) \cdot (\beta + 1) + 1$ and that all layers that do not correspond to a β-bounded (s, z)-path containing all elements of a set in C are edgeless. Since S in not β-bounded (s, z)-separator, there must exist a β-bounded (s, z)-path P' which contains two consecutive time-edges (e_1, t_1) and (e_2, t_2) where $t_1 \neq t_2$. From the construction of G, we know that there are $i < i' \in \{1, \ldots, m\}$ such that $t_1 = (i-1) \cdot (\beta + 1) + 1$ and $t_2 = (i'-1) \cdot (\beta + 1) + 1$. Without loss of generality, we assume $i + 1 = i'$. Since P' is a β-bounded (s, z)-path, the following hold:

$$\begin{aligned} t_2 - t_1 &\leqslant \beta \\ \Leftrightarrow (i'-1) \cdot (\beta + 1) + 1 - ((i-1) \cdot (\beta + 1) + 1) &\leqslant \beta \\ \Leftrightarrow ((i+1)-1) \cdot (\beta + 1) + 1 - ((i-1) \cdot (\beta + 1) + 1) &\leqslant \beta \\ \Leftrightarrow i \cdot (\beta + 1) - (i-1) \cdot (\beta + 1) &\leqslant \beta \\ \Leftrightarrow (i-1) \cdot (\beta + 1) + (\beta + 1) - (i-1) \cdot (\beta + 1) &\leqslant \beta \\ \Leftrightarrow \beta + 1 &\not\leqslant \beta \end{aligned}$$

This is a contradiction and therefore S is a β-bounded (s, z)-separator. □

Next, we show that we can apply the idea behind Reduction Rule 2.1 also in the path model of β-bounded (s, z)-path.

Reduction Rule 5.1. *Let $G = (V, E, \tau)$ be a temporal graph and let $[t_1, t_2] \subseteq [1, \tau]$ be an interval where $t_2 - t_1 \geqslant \beta - 1$ and for all $t \in [t_1, t_2]$ the layer G_t is an edgeless graph. Then for all $(\{v, w\}, t') \in E$ where $t' > t_2$, replace $(\{v, w\}, t')$ with $(\{v, w\}, t' - t_2 + t_1 - 1 + \beta)$ in E.*

Lemma 5.2. *Reduction Rule 5.1 does not remove or add any β-bounded (s, z)-path from the temporal graph $G = (V, E, \tau)$ and can be applied exhaustively in $\mathcal{O}(|E|)$ time if E is ordered by ascending labels.*

Proof. First we discuss the soundness of this rule and then the algorithm to compute it in $\mathcal{O}(|E|)$ time. Let $G = (V, E, \tau)$ be a temporal graph, $s, z \in V$, and $[t_a, t_b] \subseteq [1, \tau]$ be an interval where $t_b - t_a \geqslant \beta - 1$ and for all $t \in [t_a, t_b]$ the layer G_t is an edgeless graph. Furthermore, let $P = (e_1, t_1), \ldots, (e_i, t_i), (e_{i+1}, t_j), \ldots, (e_n, t_n)$ be a β-bounded (s, z)-path in G, and let G' be the graph after we applied Reduction Rule 5.1 on G.

Assume towards a contradiction that $t_1 < t_a$ or $t_b < t_n$. Since for all $t \in [t_a, t_b]$ the layer G_t is edgeless, we have $t_1 < t_a < t_b < t_n$. Let t' be the maximum label such that $t' < t_a$ and there is a time-edges in P which is labeled by t' and let t'' be the minimum label such that $t_b < t''$ and there is a time-edge in P which is labeled by t''. Observe that there are two consecutive time-edges $(e', t'), (e'', t'')$ in P. This contradicts P being a β-bounded (s, z)-path, because $t_b - t_a \geqslant \beta - 1 \Leftrightarrow t'' - t_a \geqslant \beta \Leftrightarrow t'' - t' > \beta$. Hence, we distinguish two cases.

Case 1: If $t_n < t_a$, then no time-edge of P is touched by Reduction Rule 5.1. Hence, P also exists in G'.

Case 2: If $t_b < t_1$, then clearly there is a β-bounded (s, z)-path $(e_1, t_1 - t_\beta + t_a - 1 + \beta), \ldots, (e_n, t_n - t_\beta + t_a - 1 + \beta)$ in G'

The other direction works analogously. We look at a β-bounded (s, z)-path in G' and compute the corresponding β-bounded (s, z)-path in G.

Reduction Rule 5.1 can be applied exhaustively by iterating over the time-edges (e_i, t_i) in the time-edge set E ordered by ascending labels, until the first t_1, t_2 with the given requirement appear. Set $x_0 := -t_2 + t_1 - 1 + \beta$. Then we iterate further over E and replace each time-edge (e, t) with $(e, t + x_0)$ until the next t_1, t_2 with the given requirement appear. Then we set $x_1 = x_0 - t_2 + t_1 - 1 + \beta$ and iterate further over E and replace each time-edge (e, t) with $(e, t + x_1)$. We repeat this procedure until the end of E is reached. Since, we iterate only once over E, this can be done in $\mathcal{O}(|E|)$ time. □

If Reduction Rule 5.1 is not applicable on a given temporal graph $G = (V, E, \tau)$, then we might have edgeless layers at the end of the interval $[1, \tau]$. In case Reduction Rule 2.2 can be applied. Recall from Lemma 2.1 that Reduction Rule 2.2 can be executed in linear time by iterating over all edges and take the maximum label as t_1, and that the vertices V and the time-edges E remain untouched by Reduction Rule 2.2. Hence, the application of Reduction Rule 2.2 does not add or remove any β-bounded (s, z)-path.

A consequence of Lemma 5.2 is that maximum label τ can be upper-bounded by the input size and β, because for each interval $[t_1, t_2] \subseteq [1, \tau]$ where $t_2 - t_1 + 1 = \beta + 1$ there is at least one edge (cf. Lemma 2.2).

Lemma 5.3. *Let $G = (V, E, \tau)$ be a temporal graph, where Reduction Rules 2.2 and 5.1 are not applicable. Then $\tau \leqslant (\beta + 1) \cdot |E|$.*

Now, we show that a β-bounded (s, z)-path can be computed in polynomial time.

Lemma 5.4. *Let $G = (V, E, \tau)$ be a temporal graph and $s, z \in V$. It is decidable in $\mathcal{O}(|V|^2 \cdot |E|)$ time whether there is a β-bounded (s, z)-path in G.*

Proof. Let $G = (V, E, \tau)$ be a temporal graph, where $V := \{s = v_1, \ldots, v_n = z\}$. We present an algorithm to solve this problem. Without loss of generality, we assume that Reduction Rules 2.2 and 5.1 are not applicable. Let $(x_1, \ldots, x_n) \in \mathbb{N}^n$. The value x_i denotes how long we can stay in vertex v_i until the we have to take another time-edge. We start with the vector $(1, 0, \ldots, 0)$. The value of x_1 will be always be one because a β-bounded (s, z)-path can have an arbitrary departure time.

We iterate over $t = 1, \ldots, \tau$. For each $i \in \{1, \ldots, n\}$ where $x_i > 0$, we perform a breadth-first search from v_i on the layer t of G. Let $R_t(v_i)$ be the set of reachable vertices from v_i in the layer t and let $R_t := \{v_r \in V \mid \exists i \in \{1, \ldots, n\} \text{ s. t. } : x_i > 0 \text{ and } v_r \in R_t(v_i)\}$.

Now, we adjust the vector. If $v_r \in R_t \setminus \{s\}$, then we set x_r to β. If $v_r \notin R_t \cup \{s\}$, then we decrease x_r by one if it is not already 0.

The next step is to increase t and repeat this procedure until $x_n > 0$ or t exceeds τ.

The running time of this algorithm is $\mathcal{O}(|V|^2 \cdot |E|)$, because we compute at most $|V|$ times a breadth-first search for each $t \in \{1, \ldots, \tau\}$. By Lemma 5.3, $\tau \in \mathcal{O}(|E|)$.

We claim that there is a $t \in \{1, \ldots, \tau\}$ where $x_n > 0$ if and only if there is a β-bounded (s, z)-path P in G.

$\Rightarrow$: This direction is easy because from the behavior of the algorithm we can conclude that v_j is reachable from s with a β-bounded (s, v_j)-path. Hence, if $x_n > 0$, then there is a β-bounded (s, z)-path because $v_n = z$.

$\Leftarrow$: Let P a β-bounded (s, z)-path in G with arrival time t_a and assume towards a contradiction that for all $t \in \{1, \ldots, \tau\}$ $x_n = 0$. Let $x_n(t)$ be the value of x_n for t. We are about to make an induction over the time-edges of P. Let $(\{s, v_{i_1}\}, t_1)$ be the first time-edge in P. Then, $x_{i_1}(t_1) = \beta$. Now let $(\{v_{i_1}, v_{i_2}\}, t_1), (\{v_{i_2}, v_{i_3}\}, t_2)$ where P visits the vertices in the following order $v_{i_1}, v_{i_2}, v_{i_3}$ and assume, induction hypothesis, that $x_{i_2}(t_1) = \beta$. Since $t_2 - t_1 \leqslant \beta$, we know that $x_{i_2}(t_2) > 0$ and hence $x_{i_3}(t_2) = \beta$. Consequently, $x_n(t_a) = \beta \neq 0$. □

One can observe that the algorithm of Corollary 3.17 solves the β-Bounded (s, z)-Separation, when algorithm of Lemma 2.4 is replaced with the algorithm of Lemma 5.4.

Corollary 5.5. *β-Bounded (s, z)-Separation can be solved in $\mathcal{O}((|V| - 2)^{|V|} \cdot |V|^2 \cdot |E|)$.*

In standard graph theory the size of a graph $G = (V, E)$ can be upper-bounded by $\mathcal{O}(|V|^2)$. Thus, the number of vertices in a graph is usually not an useful parameter. In temporal graphs the number of time-edges can be much larger than the number of vertices. This makes the number of vertices to an interesting parameter. Furthermore, we are about to show that there are instances where the input size is even exponential in the number of vertices and there does not exist a polynomial-time algorithm to decrease the instance such that there is a polynomial upper bound in the number of vertices for the input size, unless the broadly believed assumption that $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$ breaks.

Theorem 5.6. *Unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$, β-Bounded (s, z)-Separation does not admit a polynomial kernel when parameterized by the number of vertices $|V|$.*

Proof. We OR-cross-compose $p = 2^q$ instances of β-BOUNDED (s,z)-SEPARATION into one instance of β-BOUNDED (s,z)-SEPARATION.

A β-BOUNDED (s,z)-SEPARATION instance $I_i = (G_i = (V_i, E_i, \tau_i), s_i, z_i, k_i)$ is called *bad* if $k_i > |V_i|$, $\tau_i > |E_i|$, $k_i < 0$ or $\tau_i < 1$. We define the polynomial equivalence relation $\mathcal{R}$ on $I_0, \ldots, I_{p-1}$ as follows: two instances $I_i = (G_i = (V_i, E_i, \tau_i), s_i, z_i, k_i)$ and $I_j = (G_j = (V_j, E_j, \tau_j), s_j, z_j, k_j)$ are $\mathcal{R}$-equivalent if and only if $k_i = k_j$, $\tau_i = \tau_j$, and $|V_i| = |V_j|$ or both are bad. Clearly, the relation $\mathcal{R}$ fulfills condition (i) of Definition 4.27. Observe that the number of equivalence classes of finite set $J := \{I_0, \ldots, I_{p-1}\}$ is upper bounded by $|I_{\max}|^3 + 1$ where $|I_{\max}|$ is the maximum input size of an element in J, because $|V_{\max}|, |E_{\max}| \leqslant |I_{\max}|$. Hence, condition (ii) holds and $\mathcal{R}$ is a polynomial equivalence relation. Without loss of generality, we assume that there is a set $V = \{s = v_1, \ldots, v_n = z\}$ of vertices such that there is a bijective between V and V_i where $s = s_i$ and $z = z_i$ for all $i \in \{0, \ldots, p-1\}$. Thus, we consider $p = 2^q$ $\mathcal{R}$-equivalent instances $I_i = (G_i = (V, E_i, \tau), s', z', k)$ as input and OR-cross-compose into an instance $\hat{O} := (G = (\hat{V}, \hat{E}, \hat{\tau}), s, z, k')$, where $i \in \{0, \ldots, p-1\}$. For now, G, and k' are variables. Later in the construction, we will give a proper definitions for $\hat{V}$, $\hat{E}$, $\hat{\tau}$, and k'.

The overall idea is to shift all labels of an instance I_i and store the time-edges of I_i with shifted labels in $\hat{E}_i$. For each time-edge $(\{v,w\}, t) \in E_i$ we have a time-edge $(\{v,w\}, i \cdot \tau + t) \in \hat{E}_i$. Observe that $\hat{E}_i$ contains all time-edges of E_i but its labels are at least $i \cdot \tau + 1$ and at most $i \cdot \tau + \tau$. We denote the label t of E_i in $\hat{E}_i$ by $\sigma(i,t) := i \cdot \tau + t$. Therefore, $(\{v,w\}, t) \in E_i \Leftrightarrow (\{v,w\}, \sigma(i,t)) \in \hat{E}_i$. Now, we need gadgets such that all β-bounded (s,z)-paths have at least a specific departure time and at most a specific arrival time. Before we state the construction and prove its correctness, we introduce two gadgets.

Departure gadget. Now we will introduce the *departure gadget* $G_d = (V_d, E_d, \sigma(p-1,1) + \beta + 1)$. It will ensure that every β-bounded (s,z)-path in G has at least a specific departure time. In the example of Figure 5.2 the departure gadget is the upper part of the figure. The vertex set V_d consists of $q+1$ vertex sets $C_1, \ldots, C_{q+1}$, the special vertices s', z', and $2q$ *selection vertex sets* $L_{1,0}, L_{1,1}, \ldots, L_{q,0}, L_{q,1}$, where $C_1, \ldots, C_{q+1}$ contains $k'+1$ vertices and $L_{i,x}$ contains $k+1$ vertices for all $i \in \{1, \ldots, q\}$ and $x \in \{0,1\}$. Let $b(j,1), \ldots, b(j,q)$ be the binary encoding of the number $j \in \{0, \ldots, p-1\}$. Therefore, $b(j,q)$ is the least significant bit and $b(j,1)$ is the most significant bit. Let $v_1 \in C_1$ and $v_i \in C_i$. There are time-edges

$$(\{s', v_1\}, \sigma(j,1)), (\{v_i, \ell_{i,b(j,i)}\}, \sigma(j,1)), (\{\ell_{i,b(j,i)}, v_{i+1}\}, \sigma(j,1)) \in E_d,$$

where $j \in \{0, \ldots, p-1\}$ and $\ell_{i,x} \in L_{i,x}$ for all $i \in \{1, \ldots, q\}$ and $x \in \{0,1\}$. Furthermore, there are time-edges $(\{s', \ell_{i,0}\}, \sigma(p-1,1) + \beta + 1), (\{\ell_{i,0}, \ell_{i,1}\}, \sigma(p-1,1) + \beta + 1), (\{\ell_{i,1}, z'\}, \sigma(p-1,1) + \beta + 1) \in E_d$, where $\ell_{i,x} \in L_{i,x}$ for all $i \in \{1, \ldots, q\}$ and $x \in \{0,1\}$. Notice that a vertex in $L_{i,0}$ has a time-edge to all vertices in $L_{i,1}$ and vice versa, where $i \in \{1, \ldots, q\}$.

Note that $|V_d| = 2 + (q+1) \cdot (k'+1) + 2q \cdot (k+1)$ and $|E_d| = 2q \cdot (k'+1)(k+1) + 2q \cdot (k+1) + q \cdot (k+1)^2$.

One can observe that there are $k+1$ vertex-disjoint β-bounded (s', z')-paths of the form $P_i = (\{s', \ell_{i,1}\}, \sigma(p-1,1) + \beta + 1), (\{\ell_{i,0}, \ell_{i,1}\}, \sigma(p-1,1) + \beta + 1), (\{\ell_{i,0}, z'\}, \sigma(p-$

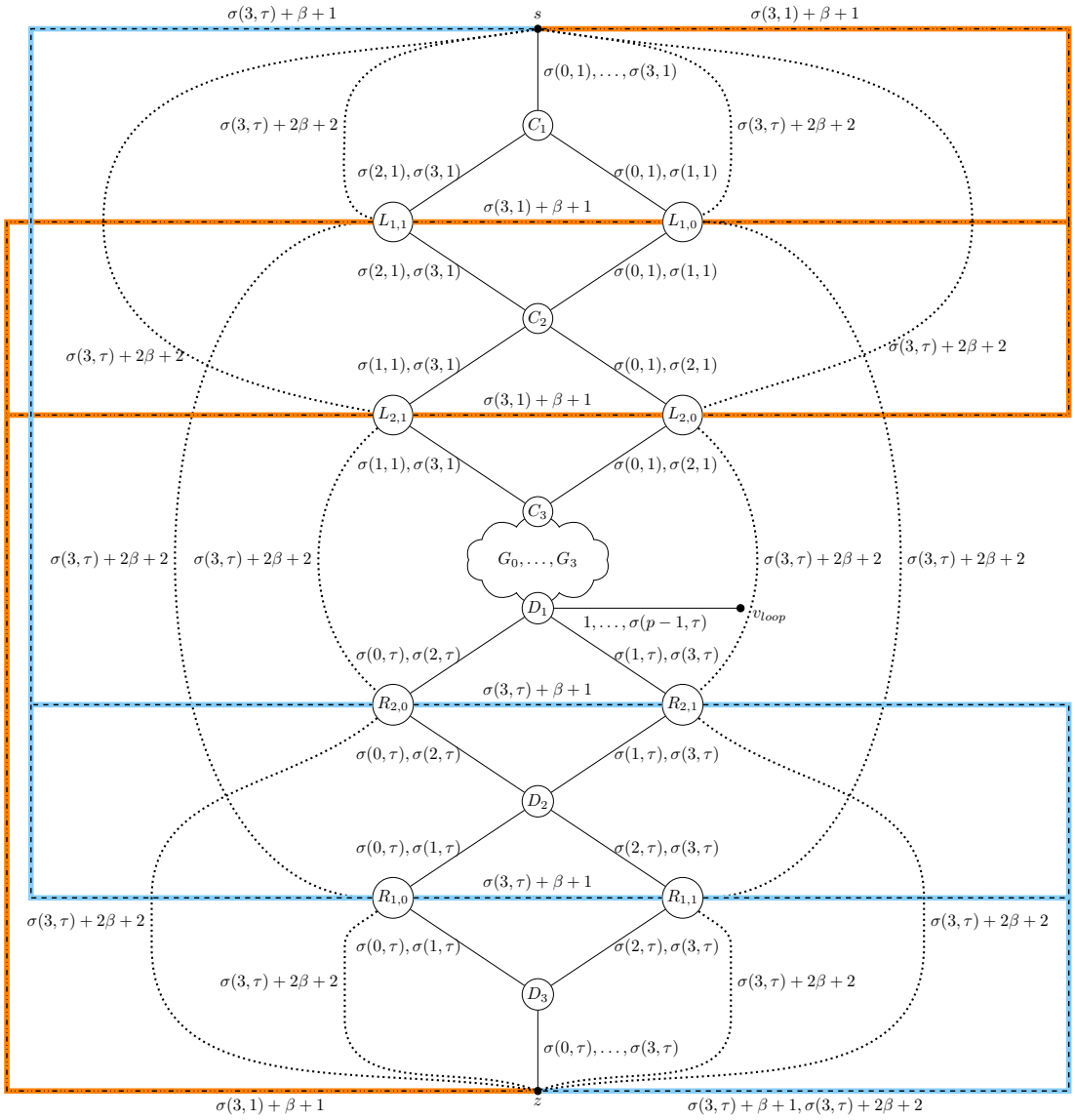


Figure 5.2: Example of the OR-cross-composition of Theorem 5.6 for $p = 4$ input instances, where time-edges between vertex sets with the same label are illustrated by one time-edge. The selector paths of the departure gadget are dash-dotted (orange), the selector path of the arrival gadget are dashed (blue) and the controller paths are dotted.

$1,1) + \beta + 1)$, where $\ell_{i,x} \in L_{i,x}$ for all $i \in \{1,\ldots,q\}$ and $x \in \{0,1\}$. We call P_i a *selector path*. In Figure 5.2, the selector paths of the departure gadget are dash-dotted (orange). Furthermore, P_j and P_i are vertex-disjoint if $i \neq j$, where $j,i \in \{1,\ldots,q\}$. Besides, $|C_i| > k'$ and therefore cannot be contained in a β-bounded (s',z')-separator of G_d. Hence, for each $i \in \{1,\ldots,q\}$ one of the selection vertex sets $L_{i,0}$ or $L_{i,1}$ must be in a β-bounded (s',z')-separator.

Moreover, there is no β-bounded (s',z')-path P' which is not equal to a P_i, where $i \in$

$\{1, \ldots, q\}$. This is the case because all time-edges which are incident with z' are labeled with $\sigma(p-1,1)+\beta+1$ and all other labels in the departure gadget are labeled with at most $\sigma(p-1,1)$. Since, $\sigma(p-1,1)+\beta+1-\sigma(p-1,1) > \beta$, no β-bounded (s', z')-path in G_d uses a time-edge labeled below $\sigma(p-1,1)+\beta+1$.

We denote the negation of $b(j,i)$ by $\overline{b(j,i)} = 1 - b(j,i)$, where $b(j,1), \ldots, b(j,q)$ is the binary encoding of $j \in \{0, \ldots, p-1\}$. Thus, there is a β-bounded (s', z')-separator $S_j^{(d)} = \bigcup_{i=1}^{q} L_{i,\overline{b(j,i)}}$ of size $q \cdot (k+1)$ of G_d and in total there are $2^q = p$ different β-bounded (s', z')-separator of size at most $q \cdot (k+1)$ in the departure gadget, where d in $S_j^{(d)}$ is just a symbol to denote that this is a β-bounded (s', z')-separator in the departure gadget G_d. Note that each of these $2^q = p$ β-bounded (s', z')-separator in G_d are of size exactly $q \cdot (k+1)$. This will be helpful to select an instance.

Claim 5.7. *Let $v_{q+1} \in C_{q+1}$ and $j \in \{0, \ldots, p-1\}$. Then $G_d - S_j^{(d)}$ admits*

(i) a β-bounded (s', v_{q+1})-path with arrival time $\sigma(j,1)$, and

(ii) no β-bounded (s', v_{q+1})-path with arrival time at most $\sigma(j,1)-1$.

For now, assume that this claim holds. We will prove it directly after we completed the current proof under the assumption that Claim 5.7 holds.

Arrival gadget. Next, we introduce the *arrival gadget* $G_a = (V_a, E_a, \sigma(p-1,\tau)+\beta+1)$. It ensures that a β-bounded (s,z)-path in G has at most a specific arrival time. In the example of Figure 5.2 the arrival gadget is the lower part of the figure. The vertex set V_a consists of $q+1$ vertex sets $D_1, \ldots, D_{q+1}$, the special vertices s', z', and $2q$ selection vertex sets $R_{1,0}, R_{1,1}, \ldots, R_{q,0}, R_{q,1}$, where D_i contains $k'+1$ vertices and $R_{i,x}$ contains $k+1$ vertices. Let $w_i \in D_i$ and $w_{q+1} \in D_{q+1}$. There are time-edges

$$(\{w_i, r_{q-(i-1),b(j,q-(i-1))}\}, \sigma(j,\tau)), (\{r_{q-(i-1),b(j,q-(i-1))}, w_{i+1}\}, \sigma(j,\tau)),$$
$$(\{w_{q+1}, z'\}, \sigma(j,\tau)) \in E_a,$$

where $r_{q-(i-1),0} \in R_{q-(i-1),0}$ and $r_{q-(i-1),1} \in R_{q-(i-1),1}$ for all $j \in \{0, \ldots, p-1\}$ and all $i \in \{1, \ldots, q\}$, Furthermore, there are time-edges $(\{s', r_{i,0}\}, \sigma(p-1,\tau)+\beta+1), (\{r_{i,0}, r_{i,1}\}, \sigma(p-1,\tau)+\beta+1), (\{r_{i,1}, z'\}, \sigma(p-1,\tau)+\beta+1) \in E_a$, where $r_{i,0} \in R_{i,0}$ and $r_{i,1} \in R_{i,1}$ for all $i \in \{1, \ldots, q\}$. Notice that a vertex in $R_{i,0}$ has a time-edge to all vertices in $R_{i,1}$ and vice versa, where $i \in \{1, \ldots, q\}$.

Note, in the contrast to the departure gadget in the arrival gadget, a β-bounded (w_1, z')-path visits at first a vertex from the selection vertex sets for the least significant bit ($R_{q,0}$ or $R_{q,1}$) and move towards the most significant bit ($R_{1,0}$ or $R_{1,1}$) The arrival gadget has the same size as the departure gadget. Hence, $|V_a| = |V_d|$ and $|E_a| = |E_d|$.

Similar to the departure gadget, we observe that there are $2^q = p$ many β-bounded (s', z')-separators of size at most $q \cdot (k+1)$ in G_a and that each of same has exactly the size of $q \cdot (k+1)$. We denote the j-th β-bounded (s', z')-separator in G_a by $S_j^{(a)} = \bigcup_{i=1}^{q} R_{i,\overline{b(j,i)}}$, where $j \in \{0, \ldots, p-1\}$.

Claim 5.8. *Let $w_1 \in D_1$ and $j \in \{0, \ldots, p-1\}$. Then $G_a - S_j^{(a)}$ admits*

(i) a β-bounded (w_1, z')-path with departure time $\sigma(j,\tau)$, and

(ii) no β-bounded (w_1, z')-path with departure time at least $\sigma(j,\tau)+1$.

For now, assume that this claim holds. We will prove it, as well as Claim 5.7, after we completed the current proof under the assumption that Claim 5.8 holds.

Construction. Next, we construct $G = (\hat{V}, \hat{E}, \hat{\tau})$. The maximum time-edge label is $\hat{\tau} = \sigma(p-1,\tau) + 2\beta + 2$ and the maximum size of any β-bounded (s', z')-separator is $k' := 2q \cdot (k+1) + k$. The vertices are $\hat{V} := V_d \cup V_a \cup (V \setminus \{s, z\}) \cup \{v_{\text{loop}}\}$. Note that $V_d \cap V_a = \{s', z'\}$ and that

$$\begin{aligned}
|\hat{V}| &= |V_a| + |V_b| - 2 + |V| - 2 + 1 \\
&= |V_a| + |V_b| + |V| - 3 \\
&= 2\left[2 + (q+1)\cdot(k'+1) + 2q\cdot(k+1)\right] + |V| - 3 \\
&= 4 + 2(q+1)\cdot(k'+1) + 4q\cdot(k+1) + |V| - 3 \\
&= 4 + 2(q+1)\cdot(2q\cdot(k+1) + k + 1) + 4q\cdot(k+1) + |V| - 3 \\
&= 2(\log(p)+1)\cdot(2\log(p)\cdot(k+1) + k + 1) + 4\log(p)\cdot(k+1) + |V| + 1 \\
&\leqslant 2(\log(p)+1)\cdot(2\log(p)\cdot(|V|+1) + |V| + 1) + 4\log(p)\cdot(|V|+1) + |V| + 1
\end{aligned}$$

Recall that for each $j \in \{0, \ldots, p-1\}$ we have a time-edge set $\hat{E}_j$ such that $(e,t) \in E_j \Leftrightarrow (e, \sigma(j,t)) \in \hat{E}_j$. Let $\hat{E}_j \setminus \{s,z\} := \{(\{v,w\}, t \in \hat{E}_j \mid v, w \in V \setminus \{s,z\}\}$. The time-edge set of G is

$$\begin{aligned}
\hat{E} := {} & \{(\{s', \ell_{i,0}\}, \hat{\tau}), (\{\ell_{i,0}, r_{i,1}\}, \hat{\tau}), (\{r_{i,1}, z'\}, \hat{\tau}) \mid \ell_{i,0} \in L_{i,0}, r_{i,1} \in R_{i,0}, i \in \{1, \ldots, q\}\} \\
& \cup \{(\{s', \ell_{i,1}\}, \hat{\tau}), (\{\ell_{i,1}, r_{i,0}\}, \hat{\tau}), (\{r_{i,0}, z'\}, \hat{\tau}) \mid \ell_{i,1} \in L_{i,1}, r_{i,0} \in R_{i,0}, i \in \{1, \ldots, q\}\} \\
& \cup \{(\{v_{q+1}, v\}, \sigma(j,t)) \mid (\{s,v\}, t) \in E_j, v_{q+1} \in C_{q+1}\} \\
& \cup \{(\{v, w_1\}, \sigma(j,t)) \mid (\{v,z\}, t) \in E_j, w_1 \in D_1\} \\
& \cup \{(\{w_1, v_{\text{loop}}\}, t) \mid w_1 \in D_1 \text{ and } t \in \{1, \ldots, \sigma(p-1,\tau)\}\} \\
& \cup \bigcup_{j=1}^{p} (\hat{E}_j \setminus \{s,z\}) \cup E_d \cup E_a
\end{aligned}$$

Here, we capture the intent of each part of $\hat{E}$. The first two time-edges sets

$$\{(\{s', \ell_{i,0}\}, \hat{\tau}), (\{\ell_{i,0}, r_{i,1}\}, \hat{\tau}), (\{r_{i,1}, z'\}, \hat{\tau}) \mid \ell_{i,0} \in L_{i,0}, r_{i,1} \in R_{i,0}, i \in \{1, \ldots, q\}\}, \text{ and}$$
$$\{(\{s', \ell_{i,1}\}, \hat{\tau}), (\{\ell_{i,1}, r_{i,0}\}, \hat{\tau}), (\{r_{i,0}, z'\}, \hat{\tau}) \mid \ell_{i,1} \in L_{i,1}, r_{i,0} \in R_{i,0}, i \in \{1, \ldots, q\}\}$$

ensure that $S_j^{(d)}$ is the β-bounded (s', z')-separator of the departure gadget G_d in G if and only if $S_j^{(a)}$ is the β-bounded (s', z')-separator of arrival gadget G_a in G. For every $i \in \{1, \ldots, q\}$ and $x \in \{0, 1\}$ there is a β-bounded (s,z)-path P_c where $V(P_c) = \{s', \ell_{i,x}, r_{i,|x-1|}, z'\}$, where $\ell_{i,x} \in L_{i,x}$ and $r_{i,|x-1|} \in R_{i,|x-1|}$. We call P_c a *controller path*. In Figure 5.2, controller paths are dotted. The sets $\{(\{v_{q+1}, v\}, \sigma(j,t)) \mid (\{s,v\}, t) \in E_j, v_{q+1} \in C_{q+1}\}$ and $\{(\{v, w_1\}, \sigma(j,t)) \mid (\{v,z\}, t) \in E_j, w_1 \in D_1\}$ replace s and z

with C_{q+1} and D_1, respectively. The sets together with $\bigcup_{j=1}^{p}(\hat{E}_j \setminus \{s,z\})$ are the time-edges of the instance $I_0,\ldots,I_{p-1}$. The sets E_d and E_a are the time-edges from the departure and arrival gadgets. Observe that all time-edges labeled with at least $\sigma(j,1)$ and at most $\sigma(j,\tau)$ correspond to the input instance I_j. The time-edge set $\{(\{w_1,v_{\text{loop}}\},t)\,|\,w_1 \in D_1 \text{ and } t \in \{1,\ldots,\sigma(p-1,\tau)\}\}$ ensures that for each β-bounded (v_{q+1},w_1)-path with arrival time t in G there is a β-bounded (v_{q+1},w_1)-path with arrival time t', where $t' \in \{t+1,\ldots,\sigma(p-1,\tau)\}$, $v_{q+1} \in C_{q+1}$ and $w_1 \in D_1$.

Note that $|\hat{E}| \in \mathcal{O}(6q\cdot(k+1)+\sigma(p-1,\tau)+|E_{\max}|+k\cdot|V|)$, where $|E_{\max}|$ is the maximum size of the time-edge set of an instance $I_0,\ldots,I_{p-1}$.

Correctness. We claim that there is a $j \in \{0,\ldots,p-1\}$ such that I_j is a yes-instance if and only if $\hat{O}$ is a yes-instance.

$\Rightarrow$: Let $j \in \{0,\ldots,p-1\}$ and $S \subseteq V \setminus \{s',z'\}$ be a β-bounded (s,z)-separator of size at most k in G_j. Set $S' := S \cup S_j^{(d)} \cup S_j^{(a)}$. Clearly, $|S'| \leqslant 2q\cdot(k+1)+k$. We already know that $S_j^{(d)}$ is a β-bounded (s',z')-separator of G_d in G and that $S_j^{(a)}$ is a β-bounded (s',z')-separator of G_a in G. Observe that no controller path exist in $G-S'$ because of $S_j^{(d)}$ and $S_j^{(a)}$. From Claim 5.7, we know that there is a β-bounded (s',v_{q+1})-path in $G-S'$ with arrival time $\sigma(j,1)$ and there is no β-bounded (s',v_{q+1})-path in $G-S'$ with arrival time at most $\sigma(j,1)-1$, where $v_{q+1} \in C_{q+1}$. From Claim 5.8, we know that there is a β-bounded (w_1,z')-path in $G-S'$ with departure time $\sigma(j,\tau)$ and there is no β-bounded (w_1,z')-path in $G-S'$ with departure time at most $\sigma(j,\tau)-1$, where $w_1 \in D_1$. Since S is a β-bounded (s,z)-separator in G_j, we can conclude that there is no β-bounded (v_{q+1},w_1)-path in $G-S'$ with departure time at least $\sigma(j,1)$ and arrival time at most $\sigma(j,\tau)$. Hence, S' is a β-bounded (s',z')-separator in G and $\hat{O}$ a yes-instance.

$\Leftarrow$: Now let $I_0,\ldots,I_{p-1}$ be no-instances and assume towards a contradiction that there is a β-bounded (s',z')-separator S' of size at most k' in G. The departure gadget G_d and arrival gadget G_a is a temporal subgraph of G. Thus, no β-bounded (s',z')-path in G_d-S' and G_a-S' exist. Hence, there are $j,j' \in \{0,\ldots,p-1\}$ such hat $S_j^{(d)}, S_{j'}^{(a)} \subseteq S'$. Let P_c be a controller path for an $i \in \{1,\ldots,q\}$ such that $V(P_c)=\{s',\ell_{i,x},r_{i,|x-1|},z'\}$, where $\ell_{i,x} \in L_{i,x}$ and $r_{i,|x-1|} \in R_{i,|x-1|}$ for $x \in \{0,1\}$. Therefore, either $\ell_{i,x} \in S'$ or $r_{i,|x-1|} \in S$. Since there is a controller path for every vertex pair in $L_{i,x} \times R_{i,|x-1|}$, either $L_{i,x} \subseteq S'$ or $R_{i,|x-1|} \subseteq S'$, where $i \in \{1,\ldots,q\}$ and $x \in \{0,1\}$. Thus, $j=j'$. Let $S = S' \setminus (S_j^{(a)} \cup S_j^{(d)})$. Observe that $|S| \leqslant k$ and hence there is no $L_{i,x}$ and no $R_{i,x}$ such that $L_{i,x} \subseteq S$ or $R_{i,x} \subseteq S$. Consequently, there is β-bounded (s',v_{q+1})-path P_1 in $G-S'$ with arrival time $\sigma(j,1)$ and there is β-bounded (w_1,z')-path P_2 with departure time $\sigma(j,\tau)$, where $v_{q+1} \in C_{q+1}$ and $w_1 \in D_1$. Since $S_j^{(d)} \cap V = \emptyset$ and $S_{j'}^{(a)} \cap V = \emptyset$, we know that S is a β-bounded (v_{q+1},w_1)-separator of G and therefore S is also a β-bounded (s,z)-separator of G_j. This contradicts I_j being a no-instance. □

It remains to be shown that the claims are correct.

Proof of Claim 5.7. We keep the notations and definitions from the proof of Theo-

rem 5.6. We prove (i) by stating the β-bounded (s', v_{q+1})-path

$$P_j = (\{s', v_1\}, \sigma(j,1)), (\{v_1, \ell_{1,b(j,1)}\}, \sigma(j,1)), (\{\ell_{1,b(j,1)}, v_2\}, \sigma(j,1)), \ldots,$$
$$(\{v_q, \ell_{q,b(j,q)}\}, \sigma(j,1)), (\{\ell_{q,b(j,q)}, v_{q+1}\}, \sigma(j,1))$$

with arrival time $\sigma(j,1)$ in $G_d - S_j^{(d)}$, where $v_i \in C_i$, $v_{q+1} \in C_{q+1}$, $\ell_{i,x} \in L_{i,x}$ for $i \in \{1,\ldots,q\}$ and $x \in \{0,1\}$. Observe that P_j visits a vertex from each selection vertex set not in $S_j^{(d)}$.

To show (ii), assume towards a contradiction that P' is a β-bounded (s', v_{q+1})-path in $G_d - S_j^{(d)}$ with arrival time at most $\sigma(j,1) - 1$, where $v_{q+1} \in C_{q+1}$. Note that all time-edges (e,t) in the time-edges set of the departure gadget E_d, either $t = \sigma(p-1,1) + \beta + 1$ or there is a $j' \in \{0, p-1\}$ such that $t = \sigma(j',1)$. Furthermore, note that there is no β-bounded (s', v_{q+1})-path with a time-edge which is also in a controller path, because all time-edges $(\{v_{q+1}, v\}, t)$ it holds that $t \leqslant \sigma(p-1,\tau)$ and all time-edges in a controller path are labeled by $\sigma(p-1,\tau) + 2\beta + 2$. Thus there is a $j' < j$ such that the arrival time of P' is $\sigma(j',1)$. We know from the construction of G_d that $V(P') \setminus (\{s', v_{q+1}\} \cup C_1 \cup \cdots \cup C_q) = L_{1,b(j,1)} \cup \cdots \cup L_{q,b(j,q)}$ and that P' must be of length $2q+1$. Let $P' = (e_1, \sigma(j'_1,1)), \ldots, (e_{2q+1}, \sigma(j'_{2q+1},1))$. Since P' is a β-bounded (s', v_{q+1})-path with arrival time $\sigma(j',1)$, we know $1 \leqslant \sigma(j'_1,1) \leqslant \cdots \leqslant \sigma(j'_{2q+1},1) = \sigma(j',1)$. The β-bounded (s', v_{q+1})-path P' visits, as well as P_j, one vertex from each selection vertex sets $L_{1,b(j,1)}, \ldots, L_{q,b(j,q)}$, because all other selection vertex sets are in the β-bounded (s', z')-separator $S_j^{(d)}$. From the construction of G_d, we know that there must be a sequence $\sigma(j'_{i_1},1) \leqslant \cdots \leqslant \sigma(j'_{i_q},1)$, where $1 \leqslant i_1 \leqslant \cdots \leqslant i_q \leqslant 2q+1$, such that the time-edge $(e_{i_u}, \sigma(j'_{i_u},1))$ is incident with $\ell_{u,b(j,u)} \in L_{u,b(j,u)}$, for all $u \in \{1,\ldots,q\}$. Note that the u-th bit of the binary encoding of j is equal to the u-th bit of the binary encoding of j'_{i_u} ($b(j,u) = b(j'_{i_u}, u)$), otherwise $(e_{i_u}, \sigma(j'_{i_u},1))$ is not incident with $\ell_{u,b(j,u)} \in L_{u,b(j,u)}$.

We are about to do an induction over q. Observe, as base case, that $j'_{i_1} < j$ and that j'_{i_1} and j have the most significant bit in common ($b(j,1) = b(j'_{i_1},1)$). Hence, $j - j'_{i_1} < 2^{q-1} - 1$. Now let $j - j'_{i_{u-1}} < 2^{q-(u-1)} - 1$. Since $j'_{i_{u-1}} \leqslant j_{i_u}$, it must hold that $j - j'_{i_u} < 2^{q-u-1} - 1$ Furthermore, we already observed that $b(j,u) = b(j'_{i_u}, u)$. Thus, $j - j'_{i_u} < 2^{q-u} - 1$. It follows that $j - j'_{i_q} < 2^{q-q} - 1 = 0$. This is a contradiction. Hence, P' does not exist. □

Proof of Claim 5.8. We keep the notations and definitions from the proof of Theorem 5.6. We prove (i) by stating the β-bounded (w_1, z')-path

$$P_j = (\{w_1, r_{q,b(j,q)}\}, \sigma(j,\tau)), (\{r_{q,b(j,q)}, w_2\}, \sigma(j,\tau)),$$
$$(\{w_2, r_{q-1,b(j,q-1)}\}, \sigma(j,\tau)), (\{r_{q-1,b(j,q-1)}, w_3\}, \sigma(j,\tau)), \ldots,$$
$$(\{w_q, r_{1,b(j,1)}\}, \sigma(j,\tau)), (\{r_{1,b(j,r)}, w_{q+1}\}, \sigma(j,\tau)),$$
$$(\{w_{q+1}, z'\}, \sigma(j,\tau))$$

with departure time $\sigma(j,\tau)$ in $G_a - S_j^{(a)}$, where $w_{q+1} \in D_{q+1}$,$w_i \in D_i$ and $r_{i,x} \in R_{i,x}$ for $i \in \{1,\ldots,q\}$ and $x \in \{0,1\}$. Observe that P_j is visiting a vertex of each selection vertex sets not in $S_j^{(a)}$.

To show (ii), assume towards a contradiction that P' is a β-bounded (w_1, z')-path in $G_a - S_j^{(a)}$ with departure time at least $\sigma(j,\tau) + 1$. Note that all time-edges (e,t) in

the time-edges set of the arrival gadget E_a, either $t = \sigma(p-1,\tau)+\beta+1$ or there is a $j' \in \{0, p-1\}$ such that $t = \sigma(j',\tau)$. Furthermore, note that there is no β-bounded (w_1, z')-path with a time-edge which is also in a controller path, because for all time-edges $(\{w_1, v\}, t)$ it holds that $t \leqslant \sigma(p-1,\tau)$ and all time-edges in a controller path are labeled by $\sigma(p-1,\tau)+2\beta+2$. Thus there is a $j' > j$ such that the departure time of P' is $\sigma(j',\tau)$. We know from the construction of G_a that $V(P')\setminus(\{z'\}\cup D_1\cup\cdots\cup D_q\cup D_{q+1}) = \bigcup_{i=1}^{q} R_{i,b(j,i)}$ and that P' must be of length $2q+1$.

Let $P' = (e_1, \sigma(j'_1,\tau)), \ldots, (e_{2q+1}, \sigma(j'_{2q+1},\tau))$. Since P' is a β-bounded (w_1, z')-path with departure time $\sigma(j',\tau)$, we know $j' \leqslant j'_1 \leqslant \cdots \leqslant j'_{2q+1}$. The β-bounded (w_1, z')-path P' visits, as well as P_j, a vertex of each selection vertex set $R_{q,b(j,q)}, \ldots, R_{1,b(j,1)}$, because all other selection vertex sets are in the β-bounded (s', z')-separator $S_j^{(a)}$.

From the construction of G_a, we know that there must be a sequence $j'_{i_1} \leqslant \cdots \leqslant j'_{i_q}$, where $1 \leqslant i_1 \leqslant \cdots \leqslant i_q \leqslant 2q+1$, such that the time-edge $(e_{i_u}, \sigma(j'_{i_u},\tau))$ is incident with $r_{q-(u-1),b(j,q-(u-1))}$, for all $u \in \{1, \ldots, q\}$. Note that $b(j, q-(u-1)) = b(j'_{i_u}, q-(u-1))$, otherwise $(e_{i_u}, \sigma(j'_{i_u},\tau))$ is not incident with $r_{q-(u-1),b(j,q-(u-1))} \in R_{q-(u-1),b(j,q-(u-1))}$ which is not according to the construction.

We are about to do an induction over q. Observe, as base case, that $j < j'_{i_q}$ and that j'_{i_q} and j have the most significant bit in common ($b(j,1) = b(j'_{i_1}, 1)$). Hence, $j'_{i_q} - j < 2^{q-1} - 1$. Now let $j'_{i_{q-((u+1)-1)}} - j < 2^{q-((u+1)-1)} - 1$. Since $j'_{i_{q-(u-1)}} \leqslant j'_{i_{q-((u+1)-1)}}$, it must hold that $j'_{i_{q-(u-1)}} - j < 2^{q-((u+1)-1)} - 1$. Furthermore, we already observed that $b(j, q-(u-1)) = b(j'_{i_u}, q-(u-1))$. Thus, $j'_{i_{q-(u-1)}} - j < 2^{q-u} - 1$. It follows that $j - j'_{i_1} < 2^{q-q} - 1 = 0$. This is a contradiction. Hence, P' does not exist. □

Chapter 6

Conclusion and Outlook

We studied the computational complexity of (Non-)Strict (s, z)-Separation and β-Bounded (s, z)-Separation with respect to the parameters solution size k, maximum label τ, underlying treewidth $\mathrm{tw}_{\downarrow}$, layer treewidth $\mathrm{tw}_{\max}$, number $|V|$ of vertices, and maximum number $|V_c|$ of vertices in a connected component over all layers.

We showed that Non-Strict (s, z)-Separation is polynomial-time solvable for $\tau = 1$ and turns NP-hard if $\tau \geqslant 2$, while Strict (s, z)-Separation is polynomial-time solvable for $\tau \leqslant 4$ and becomes NP-hard if $\tau \geqslant 5$. However, Strict (s, z)-Separation on temporal planar graphs is fixed-parameter tractable when parameterized by τ. This result is based on a fixed-parameter algorithm for the (Non-)Strict (s, z)-Separation when parameterized by $\tau + \mathrm{tw}_{\downarrow}$.

In terms of exact solutions a fixed-parameter algorithm for (Non-)Strict (s, z)-Separation on temporal planar graphs is the best we can (presumably) hope for because we showed that this problem is NP-hard on temporal planar graphs. Here, we settled an open question of Fluschnik et al. [Flu+16] by showing that Length-Bounded (s, z)-Separation is NP-hard on planar graphs.

Algorithm 1 is a simple depth-first search tree algorithm which solves Strict (s, z)-Separation in $\mathcal{O}(\tau^{k+3} \cdot |V| + |E|)$ time. One can observe that this algorithm performs well for small values of parameter k. Hence, this algorithm could be of practical interest since in many applications the goal is to keep the value of k as small as possible.

Note that there is also a variation of Algorithm 1 which solves Non-Strict (s, z)-Separation but, unfortunately, this is not a fixed-parameter algorithm which respect to $k + \tau$. A key argument in the running time estimation of Algorithm 1 for Strict (s, z)-Separation is that the length of strict (s, z)-paths is at most τ. This does not hold for non-strict (s, z)-paths. However, we managed to upper-bound the exponential part of the running time of Algorithm 1 for Non-Strict (s, z)-Separation by the parameters $k + \tau + |V_c|$ and $|V|$. Consequently, Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by $k + \tau + |V_c|$, as well as when parameterized by $|V|$. Therefore, we think that Algorithm 1 is of practical interest, even for applications of Non-Strict (s, z)-Separation because in many settings one observes a network with a constant number of vertices over a long amount of time.

Furthermore, we spotted that Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by $k + \tau + \mathrm{tw}_{\max}$. This results is based on a framework of

Mans and Mathieson [MM14] to show fixed-parameter tractability by expressing a problem for dynamic graphs in monadic second-order logic. We transferred this framework to temporal graphs.

Moreover, (Non-)Strict (s, z)-Separation becomes fixed-parameter intractable if one drops the parameter k or τ from the latter algorithm. In particular, we showed that (Non-)Strict (s, z)-Separation is W[1]-hard when parameterized by k, even if $\mathrm{tw}_{\max} = 1$, as well as it is still NP-hard, if $\mathrm{tw}_{\max} = 1$ and $\tau = 6$.

We proved that, unless NP $\subseteq$ coNP/poly, (Non-)Strict (s, z)-Separation does not admit a polynomial kernel when parameterized by $k + \tau + \mathrm{tw}_{\downarrow} + \Delta$, where Δ is the maximum degree of the temporal graph. Note that $\mathrm{tw}_{\max} \leqslant \mathrm{tw}_{\downarrow}$.

The most general path model of temporal graphs we studied is the β-bounded path model. We showed that β-Bounded (s, z)-Separation is W[2]-hard when parameterized by k and fixed-parameter tractable when parameterized by $|V|$. But unless NP $\subseteq$ coNP/poly, β-Bounded (s, z)-Separation does not admit a polynomial kernel when parameterized by $|V|$. The latter result shows, in sharp contrast to almost every problem on standard graphs, that we can not give a polynomial bound on the number of time-edges in $|V|$ for β-Bounded (s, z)-Separation.

Future research opportunities. First, we summarize further research opportunities we have already mentioned in this work. It is open whether

- Non-Strict (s, z)-Separation is fixed-parameter tractable when parameterized by $k + \tau$,
- (Non-)Strict (s, z)-Separation is fixed-parameter tractable when parameterized by $\mathrm{tw}_{\downarrow}$,
- (Non-)Strict (s, z)-Separation admits polynomial kernel when parameterized by $|V|$, and
- Non-Strict (s, z)-Separation on temporal planar graphs is fixed-parameter tractable when parameterized by τ.

It could be fruitful to study *approximation algorithms* of (Non-)Strict (s, z)-Separation. One might have already observed that Observation 2.10 can be formulated as a τ-approximation algorithm for (Non-)Strict (s, z)-Separation. An α-approximation is a polynomial-time algorithm which produces a solution whose value is within a factor of α of an optimum solution. To the best of our knowledge this is the only approximation algorithm known for (Non-)Strict (s, z)-Separation. Notably, the closely related Length-Bounded (s, z)-Separation and Length-Bounded (s, z)-Cut have already some approximation algorithms as well as inapproximability results [Bai+10; Kol17].

In this work, we focused on the strict and non-strict path model for temporal graphs and later, in Chapter 5, we started to discuss the computational complexity of the more general model of β-bounded paths. One can extend this line of research by studying further parameters, as well as the notion of (α, β)-bounded (s, z)-paths (see Chapter 2 for a definition).

A major difficulty in designing fixed-parameter algorithms for (Non-)Strict (s, z)-Separation seems to be that there is not necessarily a structural relation between two consecutive layers. But in many applications the layers i and $i + 1$ of the temporal graph

are similar. For example, suppose that the temporal graph is a road network and an edge is not present in layer i if the corresponding street is closed for maintenance work at day i. In these applications the layers i and $i+1$ of a temporal graph differ only in few edges. Hence, one might be interested in fixed-parameter algorithms with respect to the maximum edit distance between two consecutive layers.

Kempe, Kleinberg, and Kumar [KKK02] showed that the maximum number of vertex-disjoint non-strict (s, z)-paths is equal to the minimum size of a non-strict (s, z)-separator if the underlying graph excludes a fixed minor. But there are temporal graphs which do not exclude the fixed minor of Kempe, Kleinberg, and Kumar [KKK02] and the maximum number of vertex-disjoint non-strict (s, z)-paths is equal to the minimum size of a non-strict (s, z)-separator–for example, all temporal graphs with maximum label $\tau = 1$. We would like to have an exact characterization for the class of temporal graphs where the maximum number of vertex-disjoint non-strict (s, z)-paths is equal to the minimum size of a non-strict (s, z)-separator. One could try to extend this line of research by defining a temporal minor and trying to express the class of temporal graphs where the maximum number of vertex-disjoint non-strict (s, z)-paths is equal to the minimum size of a non-strict (s, z)-separator by a finite set of temporal minors.

Finally, an aspect of separators in temporal graphs which is not studied in this work is that one might want to remove layers instead of vertices to separate two vertices of a temporal graph. We think that the minimum label cut problem studied by Dutta et al. [Dut+16], Fellows, Guo, and Kanj [FGK10], and Zhang et al. [Zha+11] is related to this scenario.

Literature

[ACP87] S. Arnborg, D. G. Corneil, and A. Proskurowski. „Complexity of finding embeddings in a k-tree“. *SIAM Journal on Algebraic Discrete Methods* 8.2 (1987), pp. 277–284 (cit. on p. 54).

[Akr+15] E. C. Akrida, L. Gąsieniec, G. B. Mertzios, and P. G. Spirakis. „On temporally connected graphs of small cost“. *Proceedings of the 13th International Workshop on Approximation and Online Algorithms (WAOA '15)*. Springer. 2015, pp. 84–96 (cit. on pp. 12, 24).

[Akr+17] E. C. Akrida, J. Czyzowicz, L. Gąsieniec, Ł. Kuszner, and P. G. Spirakis. „Temporal flows in temporal networks“. *Proceedings of the 10th International Conference on Algorithms and Complexity (CIAC '17)*. Springer. 2017, pp. 43–54 (cit. on pp. 12, 15, 24, 26, 89).

[Bai+10] G. Baier, T. Erlebach, A. Hall, E. Köhler, P. Kolman, O. Pangrác, H. Schilling, and M. Skutella. „Length-bounded cuts and flows“. *ACM Transactions on Algorithms (TALG)* 7.1 (2010), 4:1–4:27 (cit. on pp. 34, 37, 65, 102).

[Bak94] B. S. Baker. „Approximation algorithms for NP-complete problems on planar graphs“. *Journal of the ACM (JACM)* 41.1 (1994), pp. 153–180 (cit. on p. 85).

[Ber96] K. A. Berman. „Vulnerability of scheduled networks and a generalization of Menger's Theorem“. *Networks* 28.3 (1996), pp. 125–134 (cit. on pp. 12, 14, 26, 30, 32).

[BJK14] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. „Kernelization lower bounds by cross-composition“. *SIAM Journal on Discrete Mathematics* 28.1 (2014), pp. 277–305 (cit. on p. 81).

[Boc+14] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin. „The structure and dynamics of multilayer networks“. *Physics Reports* 544.1 (2014), pp. 1–122 (cit. on p. 12).

[Bod+09] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. „On problems without polynomial kernels“. *Journal of Computer and System Sciences* 75.8 (2009), pp. 423–434 (cit. on p. 80).

[Bod96] H. L. Bodlaender. „A linear-time algorithm for finding tree-decompositions of small treewidth“. *SIAM Journal on Computing* 25.6 (1996), pp. 1305–1317 (cit. on p. 54).

[Cai+97] L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. „Advice classes of parameterized tractability“. *Annals of Pure and Applied Logic* 84.1 (1997), pp. 119–138 (cit. on p. 21).

[Cas+12] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. „Time-varying graphs and dynamic networks“. *International Journal of Parallel, Emergent and Distributed Systems* 27.5 (2012), pp. 387–408 (cit. on p. 12).

[CE12] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-order Logic: a Language-theoretic Approach*. Vol. 138. Cambridge University Press, 2012 (cit. on pp. 66, 68).

[Chi+16] R. Chitnis, M. Cygan, M. Hajiaghayi, M. Pilipczuk, and M. Pilipczuk. „Designing FPT algorithms for cut problems using randomized contractions“. *SIAM Journal on Computing* 45.4 (2016), pp. 1171–1229 (cit. on pp. 15, 50).

[Cor+09] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT press, 2009 (cit. on p. 46).

[Cou06] B. Courcelle. „The monadic second-order logic of graphs XVI: Canonical graph decompositions“. *Logical Methods in Computer Science* 2 (2006), pp. 1–46 (cit. on p. 68).

[Cou90] B. Courcelle. „The monadic second-order logic of graphs. I. Recognizable sets of finite graphs“. *Information and Computation* 85.1 (1990), pp. 12–75 (cit. on p. 68).

[CR01] D. G. Corneil and U. Rotics. „On the relationship between clique-width and treewidth“. *Proceedings of the 27th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '01)*. Springer Berlin Heidelberg, 2001, pp. 78–90 (cit. on p. 66).

[Cyg+11] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. van Rooij, and J. O. Wojtaszczyk. „Solving connectivity problems parameterized by treewidth in single exponential time“. *Proceedings of the 52nd annual Symposium on Foundations of Computer Science (FOCS '11)*. IEEE. 2011, pp. 150–159 (cit. on p. 65).

[Cyg+13] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk. „On multiway cut parameterized above lower bounds“. *ACM Transactions on Computation Theory (TOCT)* 5.1 (2013), 3:1–3:11 (cit. on pp. 15, 50).

[Cyg+15] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015 (cit. on pp. 21, 53, 54).

[DF13] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013 (cit. on pp. 21, 53, 54, 79).

[Die16] R. Diestel. *Graph Theory, 5th Edition*. Vol. 173. Graduate Texts in Mathematics. Springer, 2016 (cit. on pp. 20, 28, 48, 53).

[DK15] P. Dvořák and D. Knop. „Parametrized complexity of length-bounded cuts and multi-cuts“. *Proceedings of the 12th International Conference on Theory and Applications of Models of Computation (TAMC '15)*. Springer. 2015, pp. 441–452 (cit. on p. 65).

[Dut+16] T. Dutta, L. S. Heath, V. A. Kumar, and M. V. Marathe. „Labeled cuts in graphs“. *Theoretical Computer Science* 648 (2016), pp. 34–39 (cit. on p. 103).

[EF05] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer Science & Business Media, 2005 (cit. on p. 68).

[EFT94] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical Logic*. Undergraduate Texts in Mathematics. Springer-Verlag New York, 1994 (cit. on p. 68).

[EHK15] T. Erlebach, M. Hoffmann, and F. Kammer. „On Temporal Graph Exploration“. *Proceedings of the 42rd International Colloquium on Automata, Languages, and Programming (ICALP '15)*. Springer. 2015, pp. 444–455 (cit. on p. 12).

[Fer04] A. Ferreira. „Building a reference combinatorial model for MANETs“. *IEEE Network* 18.5 (2004), pp. 24–29 (cit. on p. 12).

[FF56] L. R. Ford and D. R. Fulkerson. „Maximal flow through a network“. *Canadian Journal of Mathematics (CJM)* 8.3 (1956), pp. 399–404 (cit. on p. 39).

[FG06] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Vol. XIV. Texts in Theoretical Computer Science. An EATCS Series. Springer Verlag, Berlin, 2006 (cit. on pp. 21, 36, 53, 54, 78, 79).

[FGK10] M. R. Fellows, J. Guo, and I. Kanj. „The parameterized complexity of some minimum label problems“. *Journal of Computer and System Sciences* 76.8 (2010), pp. 727–740 (cit. on p. 103).

[FHJ98] G. Fricke, S. T. Hedetniemi, and D. P. Jacobs. „Independence and irredundance in k-Regular Graphs“. *A Canadian Journal of Combinatorics* 49 (1998), pp. 271–279 (cit. on p. 71).

[Flu+16] T. Fluschnik, D. Hermelin, A. Nichterlein, and R. Niedermeier. „Fractals for kernelization lower bounds, with an application to length-bounded cut problems“. *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP '16)*. Vol. 55. Full version: `https://arxiv.org/abs/1512.00333`. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2016, 25:1–25:14 (cit. on pp. 5, 7, 17, 34, 65, 74, 75, 77, 80–87, 101).

[FMS09] P. Flocchini, B. Mans, and N. Santoro. „Exploration of periodically varying graphs“. *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC '09)*. Springer. 2009, pp. 534–543 (cit. on p. 12).

[Fom+10] F. V. Fomin, P. A. Golovach, D. Lokshtanov, and S. Saurabh. „Intractability of clique-width parameterizations“. *SIAM Journal on Computing* 39.5 (2010), pp. 1941–1956 (cit. on p. 66).

[FS11] L. Fortnow and R. Santhanam. „Infeasibility of instance compression and Succinct PCPs for NP“. *Journal of Computer and System Sciences* 77.1 (2011), pp. 91–106 (cit. on p. 80).

[GT11] P. A. Golovach and D. M. Thilikos. „Paths of bounded length and their cuts: Parameterized complexity and algorithms“. *Discrete Optimization* 8.1 (2011), pp. 72–86 (cit. on pp. 34, 65, 66).

[Gui11] S. Guillemot. „FPT algorithms for path-transversal and cycle-transversal problems“. *Discrete Optimization* 8.1 (2011), pp. 61–71 (cit. on pp. 15, 50).

[GW07] F. Gurski and E. Wanke. „Line graphs of bounded clique-width“. *Discrete Mathematics* 307.22 (2007), pp. 2734–2754 (cit. on p. 66).

[Har14] D. J. Harvey. „On treewidth and graph minors“. PhD thesis. Melbourne: University of Melbourne, 2014 (cit. on p. 66).

[Him+17] A.-S. Himmel, H. Molter, R. Niedermeier, and M. Sorge. „Adapting the Bron-Kerbosch algorithm for enumerating maximal cliques in temporal graphs“. *Social Network Analysis and Mining (SNAM)* 7.1 (July 2017), 35:1–35:16 (cit. on p. 12).

[HK73] J. E. Hopcroft and R. M. Karp. „An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs“. *SIAM Journal on Computing* 2.4 (1973), pp. 225–231 (cit. on p. 48).

[HS12] P. Holme and J. Saramäki. „Temporal networks“. *Physics Reports* 519.3 (2012), pp. 97–125 (cit. on pp. 12, 15).

[IWY16] Y. Iwata, M. Wahlström, and Y. Yoshida. „Half-integrality, LP-branching, and FPT algorithms“. *SIAM Journal on Computing* 4 (2016), pp. 1137–1411 (cit. on pp. 15, 50).

[Kar72] R. M. Karp. „Reducibility among combinatorial problems“. *Complexity of Computer Computations*. Springer, 1972, pp. 85–103 (cit. on pp. 31, 71).

[KKK02] D. Kempe, J. Kleinberg, and A. Kumar. „Connectivity and Inference Problems for Temporal Networks“. *Journal of Computer and System Sciences* 64.4 (2002), pp. 820–842 (cit. on pp. 12–14, 23, 24, 26, 30, 32, 103).

[Klo94] T. Kloks. *Treewidth: computations and approximations*. Vol. 842. Springer Science & Business Media, 1994 (cit. on p. 56).

[Kol17] P. Kolman. „On Algorithms for L-bounded Cut Problem“. *arXiv preprint arXiv:1705.02390* (2017) (cit. on p. 102).

[LNLP78] L. Lovász, V. Neumann-Lara, and M. Plummer. „Mengerian theorems for paths of bounded length“. *Periodica Mathematica Hungarica* 9.4 (1978), pp. 269–276 (cit. on p. 42).

[Mar06] D. Marx. „Parameterized graph separation problems“. *Theoretical Computer Science* 351.3 (2006), pp. 394–406 (cit. on pp. 15, 50).

[Men27] K. Menger. „Zur allgemeinen Kurventheorie“. ger. *Fundamenta Mathematicae* 10.1 (1927), pp. 96–115 (cit. on pp. 14, 30).

[Mer+13] G. B. Mertzios, O. Michail, I. Chatzigiannakis, and P. G. Spirakis. „Temporal network optimization subject to connectivity constraints“. *Proceedings of the 40rd International Colloquium on Automata, Languages, and Programming (ICALP '13)*. Springer. 2013, pp. 657–668 (cit. on pp. 12, 14, 24, 26).

[MG92] J. Misra and D. Gries. „A constructive proof of Vizing's Theorem“. *Information Processing Letters* 41.3 (1992), pp. 131–133 (cit. on p. 71).

[Mic16] O. Michail. „An introduction to temporal graphs: An algorithmic perspective“. *Internet Mathematics* 12.4 (2016), pp. 239–280 (cit. on pp. 12, 15, 24).

[MM14] B. Mans and L. Mathieson. „On the treewidth of dynamic graphs“. *Theoretical Computer Science* 554 (2014), pp. 217–228 (cit. on pp. 16, 17, 19, 24, 53, 54, 66–69, 102).

[MOR13] D. Marx, B. O'sullivan, and I. Razgon. „Finding Small Separators in Linear Time via Treewidth Reduction“. *ACM Transactions on Algorithms (TALG)* 9.4 (2013), 30:1–30:35 (cit. on pp. 15, 39, 50, 64, 73).

[MR14] D. Marx and I. Razgon. „Fixed-parameter tractability of multicut parameterized by the size of the cutset“. *SIAM Journal on Computing* 43.2 (2014), pp. 355–388 (cit. on pp. 15, 50).

[MS16] O. Michail and P. G. Spirakis. „Traveling salesman problems in temporal graphs“. *Theoretical Computer Science* 634 (2016), pp. 1–23 (cit. on p. 12).

[Nic+13] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora. „Graph metrics for temporal networks“. *Temporal Networks*. Springer, 2013, pp. 15–40 (cit. on p. 12).

[Nie06] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006 (cit. on pp. 21, 53, 54).

[Orl13] J. B. Orlin. „Max flows in $O(nm)$ time, or better“. *Proceedings of the 45th annual ACM Symposium on Theory of Computing (STOC '13)*. ACM. 2013, pp. 765–774 (cit. on p. 32).

[PS11] R. K. Pan and J. Saramäki. „Path lengths, correlations, and centrality in temporal networks“. *Physical Review E* 84.1 (2011), 016105:1–016105:9 (cit. on pp. 24, 89).

[Sto76] L. J. Stockmeyer. „The polynomial-time hierarchy“. *Theoretical Computer Science* 3.1 (1976), pp. 1–22 (cit. on p. 80).

[VLM16] T. Viard, M. Latapy, and C. Magnien. „Computing maximal cliques in link streams“. *Theoretical Computer Science* 609 (2016), pp. 245–252 (cit. on p. 12).

[Wil10] V. V. Williams. „Nondecreasing paths in a weighted graph or: How to optimally read a train schedule“. *ACM Trans. Algorithms* 6.4 (2010), 70:1–70:24 (cit. on pp. 23, 24).

[Wu+14] H. Wu, J. Cheng, S. Huang, Y. Ke, Y. Lu, and Y. Xu. „Path Problems in Temporal Graphs“. *The VLDB Journal* 7.9 (2014), pp. 721–732 (cit. on pp. 13, 15, 47).

[Zha+11] P. Zhang, J.-Y. Cai, L.-Q. Tang, and W.-B. Zhao. „Approximation and hardness results for label cut and related problems“. *Journal of Combinatorial Optimization* 21.2 (2011), pp. 192–208 (cit. on p. 103).