Complexity of the Temporal Shortest Path Interdiction Problem

Jan Boeckmann
TUM Campus Straubing for Biotechnology and Sustainability,
Hochschule Weihenstephan-Triesdorf, Germany

Clemens Thielen
TUM Campus Straubing for Biotechnology and Sustainability,
Hochschule Weihenstephan-Triesdorf, Germany
Department of Mathematics, School of Computation, Information and Technology,
Technische Universität München, Germany

Alina Wittmann
Department of Mathematics, School of Computation, Information and Technology,
Technische Universität München, Germany

Abstract

In the shortest path interdiction problem, an interdictor aims to remove arcs of total cost at most a given budget from a directed graph with given arc costs and traversal times such that the length of a shortest s-t-path is maximized. For static graphs, this problem is known to be strongly $\mathcal{NP}$-hard, and it has received considerable attention in the literature.

While the shortest path problem is one of the most fundamental and well-studied problems also for temporal graphs, the shortest path interdiction problem has not yet been formally studied on temporal graphs, where common definitions of a "shortest path" include: *latest start path* (path with maximum start time), *earliest arrival path* (path with minimum arrival time), *shortest duration path* (path with minimum traveling time including waiting times at nodes), and *shortest traversal path* (path with minimum traveling time *not* including waiting times at nodes).

In this paper, we analyze the complexity of the shortest path interdiction problem on temporal graphs with respect to all four definitions of a shortest path mentioned above. Even though the shortest path interdiction problem on static graphs is known to be strongly $\mathcal{NP}$-hard, we show that the latest start and the earliest arrival path interdiction problems on temporal graphs are polynomial-time solvable. For the shortest duration and shortest traversal path interdiction problems, however, we show strong $\mathcal{NP}$-hardness, but we obtain polynomial-time algorithms for these problems on extension-parallel temporal graphs.

2012 ACM Subject Classification Theory of computation → Shortest paths; Theory of computation → Dynamic graph algorithms; Mathematics of computing → Paths and connectivity problems

Keywords and phrases Temporal Graphs, Interdiction Problems, Complexity, Shortest Paths, Most Vital Arcs

Digital Object Identifier 10.4230/LIPIcs.SAND.2023.9

1 Introduction

Not least because of its great applicability to a wide range of real-world problems, the shortest s-t-path problem is undeniably one of the most central and well-studied problems in graph theory and network optimization. On static graphs, where the graph is not subject to change over time, efficient algorithms to solve the shortest path problem are known. The assumption of a graph not changing over time, however, is often too restrictive when modeling real-world problems such as the spread of the virus during the COVID-19 pandemic. In such settings, the concept of *temporal graphs*, where arcs are only available at certain times, allows for more realistic models (see, e.g., [4, 17]) and has recently attracted the interest of researchers in algorithmic network optimization (see, e.g., [1, 26, 27]).

Another topic that has become increasingly important in times of uncertainty are *interdiction problems*, where an interdictor aims to remove arcs of total cost at most a given budget from a (directed or undirected) graph or network such that the optimal objective value of an optimization problem on the resulting graph or network is maximized (in case of a minimization problem) or minimized (in case of a maximization problem). An important example of a highly relevant interdiction problem is the *shortest path interdiction problem*, where arcs are to be removed from a graph subject to a given budget such that the length of a shortest s-t-path for two given nodes s and t is maximized. Another example of a well-studied interdiction problem is the *network flow interdiction problem*, where arcs are to be removed from a network subject to a given budget such that the value of a maximum s-t-flow between two given nodes s and t is minimized [33, 37]. This problem is known to be strongly $\mathcal{NP}$-hard and several results about its approximability have been obtained [8, 11, 15].

In this paper, we investigate the *temporal shortest path interdiction problem*, where the aim is to remove arcs from a directed temporal graph such that the length of a shortest path from a node s to another node t is maximized. As the length of a path in a temporal graph can be interpreted in various different ways, we investigate four common versions of the temporal shortest path interdiction problem. We show that two of these versions are polynomial-time solvable, while the other two are strongly $\mathcal{NP}$-hard.

1.1 Previous Work

The following paragraphs summarize the state-of-the-art concerning shortest path problems on temporal graphs, the (static) shortest path interdiction problem, and related interdiction problems on temporal graphs.

We start with an overview of the literature about shortest path problems on temporal graphs. The model of a temporal graph used in this paper (and, e.g., in [10, 38]) is sometimes also referred to as a *scheduled network* [7] or a *point-availability time-dependent network* [9]. Here, each temporal arc r can only be entered at a given start time $\tau(r)$ and it takes $\lambda(r)$ units of time to traverse the arc, which leads to an arrival time of $\tau(r) + \lambda(r)$ at the end node of the arc. In this model, four different definitions of a "shortest path" between two nodes s and t are considered (see [38]):

- *reverse-foremost* or *latest start* path, which is an s-t-path with maximum start time of the first arc in the path,
- *foremost* or *earliest arrival* path, which is an s-t-path with minimum arrival time of the last arc in the path,
- *shortest duration* path, which is an s-t-path with minimum total traveling time including waiting times at the nodes,
- *shortest traversal* path, which is an s-t-path with minimum total traveling time *not* including waiting times at the nodes.

For each of the four definitions, the corresponding temporal shortest path problem can be solved efficiently, i.e., a shortest path can be computed in polynomial time [5, 10, 38].

A different definition of temporal graphs is considered, e.g, in [27], where a wide range of well-studied graph problems is investigated on temporal graphs. This definition can be interpreted as the special case of the previous definition obtained when all traversal times are zero.[1] Biobjective versions of temporal shortest path problems are considered in [9, 30, 31].

[1] This implies that all polynomial-time solvability results presented here can immediately be transferred to the definition used in [27]. For our hardness results, we point out explicitly whether they can be transferred to this definition of temporal graphs.

A definition that allows for continuous availability of arcs in a temporal graph as well as a time dependency of an arc's traversal time is provided in [13]. This definition can be seen as a generalization of the definition from [10, 38] used here. However, due to the definition's large generality, it does not allow for a finite encoding of temporal graphs without imposing further assumptions, so classical techniques of complexity analysis cannot be applied for the most general form of this definition. A natural finite encoding is possible, e.g., if each arc is restricted to be present over a time interval, i.e., it can be entered at any time between two specified points in time. Even for this special case of the definition in [13], it is shown in Section 4.1 that deciding whether two nodes s and t can be separated by removing no more than B arcs from the graph is already strongly $\mathcal{NP}$-hard.

Next, the literature about the shortest path interdiction problem on static graphs is summarized. To explicitly distinguish between the problem on *static* graphs and the problem on *temporal* graphs, we refer to the shortest path interdiction problem on static graphs as the *static* shortest path interdiction problem (S-SP-IP) in the following. This problem is also referred to as the *most vital arcs problem* in the literature [2]. S-SP-IP is one of the most-studied network interdiction problems and a vast amount of literature exists on the problem. A detailed overview is provided in [35]. Concerning the complexity of S-SP-IP, the first proof of weak $\mathcal{NP}$-hardness is provided in [2]. This result is extended in [3], where it is shown that S-SP-IP is strongly $\mathcal{NP}$-hard even on acyclic graphs and for the special case of unit arc lengths and removal costs. This result is further extended in [24], where it is shown that it is $\mathcal{NP}$-hard to approximate S-SP-IP within any factor $\alpha < 2$. Indeed, it is still an open question whether any non-trivial approximation algorithms exist for S-SP-IP. Variations of S-SP-IP considering online settings, randomized interdiction strategies, or multiple objectives have recently been studied, e.g., in [12, 21, 34].

While, to the best of our knowledge, the complexity of the shortest path interdiction problem has not been formally investigated on temporal graphs, a polynomial-time algorithm that decides whether there exist k arc-disjoint temporal s-t-paths is presented in [7]. They further show that it can be decided in polynomial time whether there exists a temporal s-t-path arriving before a given arrival time even if up to k arcs are removed, which implicitly solves the temporal earliest arrival path interdiction problem for unit removal costs. However, it is not clear whether the algorithm can be extended to the case in which arcs can have different removal costs.

Further, related reachability interdiction problems on temporal graphs are studied in [16, 17, 18, 29]. Here, the goal is to minimize (or maximize in some cases) the number of nodes reachable from a single node or a set of nodes in a temporal graph by either removing arcs, delaying start times, or changing the order of start times. While the vast majority of studied problems turn out to be $\mathcal{NP}$-hard even under severe restrictions, only a few special cases are shown to be polynomial-time solvable. Moreover, the problem of separating two given nodes by removing nodes from a temporal graph is considered for various settings in [20, 22, 23, 25, 28, 39]. Again, most of the problems are $\mathcal{NP}$-hard, while some polynomial-time solvability results – mostly for specific classes of graphs – are shown.

1.2 Our Contribution

We analyze the complexity of the shortest path interdiction problem on temporal graphs with respect to all four definitions of a shortest path considered in [38]. Even though S-SP-IP is known to be strongly $\mathcal{NP}$-hard, it is found that polynomial-time algorithms for the latest start and the earliest arrival path interdiction problem on temporal graphs exist. These algorithms exploit the fact that, for these versions of the problem, the objective value of a

path only depends on either the first or on the last arc in the path (but not on both). For the shortest duration and shortest traversal path interdiction problem, where both the first and the last arc in a path (and the amount of time spend waiting at nodes in the former case) are relevant for its length, however, we show strong $\mathcal{NP}$-hardness. Our reduction further implies that, unless $\mathcal{P} = \mathcal{NP}$, there exist no polynomial-time approximation algorithms with approximation ratio smaller than $3/2$ for these problems.

On extension-parallel temporal graphs, however, we obtain polynomial-time algorithms for the shortest duration path interdiction problem and the shortest traversal path interdiction problem. This result can be transferred to the static shortest path interdiction problem, where it also represents a new result.

2 Problem Definition

A directed (discrete-time) *temporal graph* G consists of a nonempty, finite set V of nodes and a finite set R of *temporal arcs*. As usual, we denote the number of nodes and the number of (temporal) arcs in the graph by n and m, respectively. A temporal arc $r \in R$ has four attributes, namely its *start node* $\alpha(r) \in V$, its *end node* $\omega(r) \in V$, its *start time* $\tau(r) \in \mathbb{Q}$, and its *traversal time* $\lambda(r) \in \mathbb{Q}_{\geq 0}$. When traversing a temporal arc $r \in R$, the *arrival time* of r is $\tau(r) + \lambda(r)$. A *temporal path* $P = (r_1, \dots, r_k)$ is a sequence of temporal arcs such that, for each $i \in \{1, \dots, k-1\}$, it holds that $\omega(r_i) = \alpha(r_{i+1})$ and $\tau(r_i) + \lambda(r_i) \leq \tau(r_{i+1})$, i.e., the end node of each arc is the start node of the next arc in the path and the arrival time of each arc is less than or equal to the start time of the next arc.[2] For two nodes $s, t \in V$, a temporal path $P = (r_1, \dots, r_k)$ is called a *(temporal) s-t-path* if $\alpha(r_1) = s$ and $\omega(r_k) = t$. Given a temporal graph $G = (V, R)$, the *underlying static graph* $G^{\text{stat}} = (V^{\text{stat}}, R^{\text{stat}})$ is the (directed) static graph with the same nodes and arcs obtained by disregarding the start times and traversal times of the arcs. A temporal graph is called *acyclic* if its underlying static graph is acyclic, i.e., its underlying static graph does not contain any directed cycle.

While the notion of a "shortest" s-t-path is straightforward in static graphs, temporal graphs allow for various interpretations of the term "shortest". In this paper, we study the four quality measures for s-t-paths that are used in [38].

▶ **Definition 1.** *Let G be a temporal graph, $s \neq t$ two nodes in G, and $P = (r_1, \dots, r_k)$ a temporal s-t-path.*

- *The* start time *of P is defined as* $\text{start}(P) \coloneqq \tau(r_1)$.
- *The* arrival time *of P is defined as* $\text{arriv}(P) \coloneqq \tau(r_k) + \lambda(r_k)$.
- *The* duration *of P is defined as* $\text{dura}(P) \coloneqq \text{arriv}(P) - \text{start}(P)$.
- *The* traversal time *of P is defined as* $\text{trav}(P) \coloneqq \sum_{i=1}^{k} \lambda(r_i)$.

▶ **Definition 2.** *Let G be a temporal graph and $s \neq t$ two nodes in G.*

- *A* latest start path *is an s-t-path with maximum start time. The* latest start time *in G, denoted by* $\text{LS}(G)$*, is defined as the start time of a latest start path in G.*
- *An* earliest arrival path *is an s-t-path with minimum arrival time. The* earliest arrival time *in G, denoted by* $\text{EA}(G)$*, is defined as the arrival time of an earliest arrival path in G.*

[2] Note that this definition allows a path to visit the same node (or even traverse the same arc) several times. Except for some results obtained in the setting with waiting time constraints considered in Section 4.2, however, all our results also hold when restricting to *simple* paths that do not visit any node more than once.

- *A* shortest duration path *is an s-t-path with minimum duration. The* shortest duration *in G, denoted by* $\mathrm{SD}(G)$*, is defined as the duration of a shortest duration path in G.*
- *A* shortest traversal path *is an s-t-path with minimum traversal time. The* shortest traversal time *in G, denoted by* $\mathrm{ST}(G)$*, is defined as the traversal time of a shortest traversal path in G.*

If no s-t-path exists in G, $\mathrm{LS}(G)$ *is set to* $-\infty$*, whereas* $\mathrm{EA}(G)$*,* $\mathrm{SD}(G)$*, and* $\mathrm{ST}(G)$ *are set to* $+\infty$*.*

Note that earliest arrival paths are called foremost paths and latest start paths are called reverse-foremost paths in [38]. Next, the four versions of the temporal shortest path interdiction problem are defined.

▶ **Definition 3.** *For an objective* $\mathrm{OBJ} \in \{\mathrm{LS}, \mathrm{EA}, \mathrm{SD}, \mathrm{ST}\}$*, the temporal* OBJ *path interdiction problem (T-*OBJ*P-IP) is defined as follows.*

INSTANCE:	*A temporal graph $G = (V, R)$, two nodes $s \neq t$ in G, a budget $B \in \mathbb{Q}_{>0}$, and removal costs $c : R \to \mathbb{Q}_{\geq 0}$*
TASK:	*Find a subset $S \subseteq R$ of arcs with $\sum_{r \in S} c(r) \leq B$ such that $\mathrm{OBJ}(G_S)$ is maximized (minimized in the case that $\mathrm{OBJ} = \mathrm{LS}$), where $G_S \coloneqq (V, R \setminus S)$.*

*A solution $S \subseteq R$ of T-*OBJ*P-IP with $\sum_{r \in S} c(r) \leq B$ is called an* interdiction strategy *and the arcs in S are called* interdicted*. Further, if no temporal path from a node u to another node v exists after the arcs in S have been removed, we say that the interdiction strategy S* separates *u from v or that the pair (u, v) is* separated *by S.*

3 Polynomial-Time Algorithms and Complexity Results

In this section, we analyze the complexity of each of the four introduced versions of temporal shortest path interdiction. It is shown that two versions can be solved in polynomial time and the other two versions are strongly $\mathcal{NP}$-hard. On extension-parallel temporal graphs, however, the two hard versions are shown to be solvable in polynomial time.

3.1 Temporal Latest Start Path Interdiction

In this section, we present a polynomial-time algorithm to solve T-LSP-IP. This is a surprising result as the static shortest path interdiction problem is known to be strongly $\mathcal{NP}$-hard [3]. The main reason for the polynomial-time solvability of T-LSP-IP is that the obtained objective value only depends on the first arc that is used by a latest start path in the interdicted graph G_S.

In this section, we let $\tau_1 < \tau_2 < \cdots < \tau_l$ denote the distinct start times of outgoing arcs of s in G sorted in increasing order. Further, for $k \in \{1, \ldots, l\}$, we define $G^{\mathrm{LS},k}$ as the temporal graph that results from G by removing all outgoing arcs of s with start time at most τ_k. For completeness, we also define $G^{\mathrm{LS},0} \coloneqq G$. Our algorithm is based on the following proposition.

▶ **Proposition 4.** *Let $k \in \{1, \ldots, l\}$. There exists an interdiction strategy S^k that separates s from t in $G^{\mathrm{LS},k}$ if and only if there exists an interdiction strategy S in G with objective value at most τ_k.*

Proof. Let S^k be an interdiction strategy that separates s from t in $G^{\mathrm{LS},k}$. Then, after interdicting the same set $S := S^k$ of arcs in G, no s-t-path in G_S can start with an arc with start time strictly larger than τ_k (otherwise, the path would also be an s-t-path in $G^{\mathrm{LS},k}_{S^k}$). Hence, all s-t-paths in G_S have start time at most τ_k, i.e., S has objective value at most τ_k.

Conversely, let S be an interdiction strategy in G with objective value at most τ_k. Then, no s-t-path in G_S can have start time strictly larger than τ_k, so the interdiction strategy $S^k := S \cap R^{\mathrm{LS},k}$, where $R^{\mathrm{LS},k}$ is the arc set of $G^{\mathrm{LS},k}$, separates s from t in $G^{\mathrm{LS},k}$. ◀

The idea of the algorithm is to use binary search in order to find $k^\star \in \{1, \dots, l\}$ such that s can be separated from t in $G^{\mathrm{LS},k^\star}$, but s cannot be separated from t in $G^{\mathrm{LS},k^\star-1}$. Such a $k^\star$ exists whenever s cannot already be separated from t in the whole graph $G = G^{\mathrm{LS},0}$, i.e., whenever the optimal objective value is not equal to $-\infty$. Consequently, in order to obtain a polynomial-time algorithm for T-LSP-IP, it only remains to show that deciding whether a node s can be separated from another node t with a given interdiction budget in an arbitrary temporal graph is possible in polynomial time.

In a static graph, this question can be answered easily by computing a minimum s-t-cut and comparing its cost to the given interdiction budget B. Hence, we now describe how the question in an arbitrary temporal graph $H = (V, R)$ can be reduced to the static case. To this end, we use a graph construction that is similar to [38] and to the construction of time-expanded networks in the context of dynamic flows [32]. The constructed graph is therefore called the *time-expanded graph* of H and denoted by $H^{\mathrm{te}} = (V^{\mathrm{te}}, R^{\mathrm{te}})$. We start by defining the set of *crucial times* by $T := \cup_{r \in R}\{\tau(r), \tau(r) + \lambda(r)\}$. For easier notation, we write $T = \{\phi_1, \dots, \phi_j\}$, where the crucial times are indexed in increasing order. For each $v \in V$ and $\phi \in T$, there exists a node (v, ϕ) in V^{te}. For each $i \in \{1, \dots, j-1\}$ and for each $v \in V$, there exists an arc from (v, ϕ_i) to (v, ϕ_{i+1}) with removal cost $B + 1$ (i.e., it cannot be interdicted). This arc represents waiting at node v of the temporal graph until the next crucial time. Further, for each arc $r \in R$, there exists an arc in R^{te} from $(\alpha(r), \tau(r))$ to $(\omega(r), \tau(r) + \lambda(r))$ with removal cost $c(r)$, which represents traversing arc r in the temporal graph. We define $s^{\mathrm{te}} := (s, \phi_1)$ and $t^{\mathrm{te}} := (t, \phi_j)$. If the temporal graph H has n nodes and m arcs, its time-expanded graph has $n \cdot |T| \in \mathcal{O}(n \cdot m)$ nodes and $n \cdot (|T| - 1) + m \in \mathcal{O}(n \cdot m)$ arcs. Hence, the size of the time-expanded graph is polynomial in the size of the temporal graph (in contrast to time-expanded networks used in the context of dynamic flows).
The following observation follows directly from the construction of H^{te}.

▶ **Observation 5.** *There exists an interdiction strategy separating s from t in H if and only if there exists an interdiction strategy separating s^{te} from t^{te} in H^{te}.*

Applying the previously described algorithm together with Proposition 4 and Observation 5 yields the main theorem of this section.

▶ **Theorem 6.** *There exists a polynomial-time algorithm for T-LSP-IP with running time in $\mathcal{O}(\log(l) \cdot T_{\mathrm{MC}}(n \cdot m, n \cdot m))$, where $l \leq m$ is the number of distinct start times of outgoing arcs of s and $T_{\mathrm{MC}}(n \cdot m, n \cdot m)$ is the time required to compute a minimum s-t-cut in a static graph with $n \cdot m$ nodes and $n \cdot m$ arcs.*

3.2 Temporal Earliest Arrival Path Interdiction

In this section, we present a polynomial-time algorithm to solve T-EAP-IP. Similar to T-LSP-IP, the reason for the problem's polynomial-time solvability is that the obtained objective value only depends on the last arc that is used by an earliest arrival path in the interdicted

graph G_S. Indeed, an instance of T-EAP-IP can be transformed into an equivalent instance of T-LSP-IP by inverting the direction of all arcs and adjusting the start times and traversal times appropriately. This is described in the following.

Let $G = (V, R)$ be the temporal graph in an instance of T-EAP-IP. We construct a graph $G^{\mathrm{LS}} = (V, R^{\mathrm{LS}})$ for an instance of T-LSP-IP. The *maximum arrival time* in G is defined as $\Phi := \max_{r \in R} \tau(r) + \lambda(r)$. For each $r \in R$, an arc r' is added to R^{LS} with $\alpha(r') := \omega(r)$, $\omega(r') := \alpha(r)$, $\tau(r') := \Phi - \tau(r) - \lambda(r)$, and $\lambda(r') := \lambda(r)$. The arcs r and r' are called *associated*. Further, an interdiction strategy S in G and the interdiction strategy S' in G^{LS} consisting of the arcs in G^{LS} that are associated with those in S are also called *associated*. Defining $s^{\mathrm{LS}} := t$ and $t^{\mathrm{LS}} := s$, $B^{\mathrm{LS}} := B$, and $c^{\mathrm{LS}}(r') := c(r)$ for each pair of associated arcs r and r', it is then easy to see that mapping an interdiction strategy S in G to its associated interdiction strategy S' in G^{LS} defines a bijection between the sets of interdiction strategies in the two graphs. In the following, the instance of T-EAP-IP is denoted by (G, s, t) and the constructed instance of T-LSP-IP by $(G^{\mathrm{LS}}, s^{\mathrm{LS}}, t^{\mathrm{LS}})$. We can then show the following one-to-one correspondence between temporal paths in G and G^{LS}.

▶ **Proposition 7.** *Let r_i and r'_i be associated arcs for each $i \in \{1, \dots, k\}$, and let S and S' be associated interdiction strategies in G and G^{LS}, respectively. Then $P' = (r'_1, \dots, r'_k)$ is a temporal s^{LS}-t^{LS}-path in $G^{\mathrm{LS}}_{S'}$ if and only if $P = (r_k, \dots, r_1)$ is a temporal s-t-path in G_S.*

Proof. If $P' = (r'_1, \dots, r'_k)$ is a temporal s^{LS}-t^{LS}-path in $G^{\mathrm{LS}}_{S'}$, then $r'_i \notin S'$ for $i = 1, \dots, k$. Hence, since S' and S are associated, we obtain that $r_i \notin S$ for $i = 1, \dots, k$. Moreover, $t = s^{\mathrm{LS}} = \alpha(r'_1) = \omega(r_1)$, $s = t^{\mathrm{LS}} = \omega(r'_k) = \alpha(r_k)$, and for each $i \in \{1, \dots, k-1\}$, we have $\alpha(r_i) = \omega(r'_i) = \alpha(r'_{i+1}) = \omega(r_{i+1})$ and

$$\tau(r_{i+1}) + \lambda(r_{i+1}) = \Phi - \tau(r'_{i+1}) \leq \Phi - \tau(r'_i) - \lambda(r'_i) = \Phi - \tau(r'_i) - \lambda(r_i) = \tau(r_i).$$

Thus, $P = (r_k, \dots, r_1)$ is a temporal s-t-path in G_S as claimed. The inverse direction can be shown along the same lines. ◀

We call paths P and P' as in Proposition 7 *associated* in the following. Proposition 7 allows us to show the following relationship between the objective values of associated interdiction strategies for (G, s, t) and $(G^{\mathrm{LS}}, s^{\mathrm{LS}}, t^{\mathrm{LS}})$.

▶ **Corollary 8.** *An interdiction strategy S for (G, s, t) has objective value v if and only if the associated interdiction strategy S' for $(G^{\mathrm{LS}}, s^{\mathrm{LS}}, t^{\mathrm{LS}})$ has objective value $\Phi - v$.*

Proof. Given an interdiction strategy S with objective value v and its associated interdiction strategy S', let P be an earliest arrival path in G_S. Then, P has arrival time v and by Proposition 7, the associated path P' is a temporal path in $G^{\mathrm{LS}}_{S'}$, whose start time is $\Phi - v$. For the sake of a contradiction, suppose that there exists a path $\bar{P}'$ in $G^{\mathrm{LS}}_{S'}$ with start time $\Phi - \bar{v} > \Phi - v$. Then, by Proposition 7, the path $\bar{P}$ that is associated to $\bar{P}'$ is a temporal path in G_S and its arrival time is $\bar{v} < v$, which is a contradiction to P being an earliest arrival path in G_S. Hence, the interdiction strategy S' for $(G^{\mathrm{LS}}, s^{\mathrm{LS}}, t^{\mathrm{LS}})$ has objective value $\Phi - v$. The inverse direction can be shown along the same lines. ◀

Corollary 8 immediately yields the following result.

▶ **Corollary 9.** *An interdiction strategy S is optimal for (G, s, t) if and only if its associated interdiction strategy S' is optimal for $(G^{\mathrm{LS}}, s^{\mathrm{LS}}, t^{\mathrm{LS}})$.*

Corollary 9 and the algorithm presented in Section 3.1 yield the main result of this section.

▶ **Theorem 10.** *There exists a polynomial-time algorithm for T-EAP-IP with running time in* $\mathcal{O}(\log(l) \cdot T_{\mathrm{MC}}(n \cdot m, n \cdot m))$*, where* $l \leq m$ *is the number of distinct arrival times of incoming arcs of* t *and* $T_{\mathrm{MC}}(n \cdot m, n \cdot m)$ *is the time required to compute a minimum* s*-*t*-cut in a static graph with* $n \cdot m$ *nodes and* $n \cdot m$ *arcs.*

3.3 Temporal Shortest Duration Path Interdiction and Temporal Shortest Traversal Path Interdiction

In this section, we show that T-SDP-IP and T-STP-IP are strongly $\mathcal{NP}$-hard, even for unit removal costs and if the underlying static graph is acyclic. Moreover, the reduction implies an inapproximability result. We also show, however, that both problems are solvable in polynomial time if the graph is extension-parallel. This result is also shown for the static problem S-SP-IP.

The proof of strong $\mathcal{NP}$-hardness is similar to the proof in [6], where it is shown that finding a multicut in directed acyclic graphs is $\mathcal{APX}$-hard. The reduction is performed from the strongly $\mathcal{NP}$-hard MAX2SAT problem, which is defined as follows.

INSTANCE:	A set $X = \{x_1, \ldots, x_\zeta\}$ of boolean variables, a set $C = \{c_1, \ldots, c_\mu\}$ of clauses each containing two literals, and a positive integer $\delta < \mu$
QUESTION:	Is there a truth assignment for the variables that satisfies at least δ clauses?

Given an instance of MAX2SAT, we construct a temporal graph with removal costs and a corresponding budget. This graph has the property that no s-t-path waits in any node except for s and t, which means that, for each feasible interdiction strategy, the objective values in T-SDP-IP and T-STP-IP are identical. Hence, the resulting instances of T-SDP-IP and T-STP-IP are equivalent in this case. Thus, we present the construction and the corresponding proofs only for T-SDP-IP in the following. An example for the construction is provided in Figure 1.

Unless explicitly stated otherwise, all arcs within this construction have start time 0, traversal time 0, and removal cost $B + 1$ (i.e., they cannot be interdicted). We show later that only a slight modification of the construction is necessary in the case of unit removal costs. For each variable $x_i \in X$, there is a *variable gadget* consisting of a directed path with trace $(u_{i,1}, u_{i,2}, u_{i,3}, u_{i,4})$ where only the arcs from $u_{i,1}$ to $u_{i,2}$ and from $u_{i,3}$ to $u_{i,4}$ can be interdicted at a removal cost of $N \coloneqq \mu + 1$. Interdicting the arc from $u_{i,1}$ to $u_{i,2}$ is later identified with setting x_i to true and interdicting the arc from $u_{i,3}$ to $u_{i,4}$ is identified with setting x_i to false. For each clause $c_j \in C$, there is a *clause gadget* consisting of a directed path with trace $(v_{j,1}, v_{j,2}, v_{j,3}, v_{j,4})$ where only the arcs from $v_{j,1}$ to $v_{j,2}$ and from $v_{j,3}$ to $v_{j,4}$ can be interdicted at a removal cost of one.

We next describe the arcs that connect the variable gadgets to the clause gadgets. For a clause $c_j = \hat{x}_i \vee \hat{x}_k$, where $\hat{x}_i \in \{x_i, \overline{x}_i\}$ and $\hat{x}_k \in \{x_k, \overline{x}_k\}$, we call $\hat{x}_i$ the *first literal* and $\hat{x}_k$ the *second literal* of clause c_j. For each clause, arcs are then added as follows depending on its first and second literal: If the first literal of clause c_j is x_i ($\overline{x}_i$), there exists an arc from $u_{i,2}$ to $v_{j,1}$ (from $u_{i,4}$ to $v_{j,1}$). If the second literal of clause c_j is x_k ($\overline{x}_k$), there exists an arc from $u_{k,2}$ to $v_{j,3}$ (from $u_{k,4}$ to $v_{j,3}$).

The construction is continued by adding another six nodes s_1, s_2, s_3, t_1, t_2, and t_3 to the graph. For each $i \in \{1, \ldots, \zeta\}$, there exists an arc from s_1 to $u_{i,1}$ and an arc from $u_{i,4}$ to t_1. For each $i \in \{1, \ldots, \zeta\}$, there exists an arc from s_2 to $u_{i,1}$ and another arc from s_2 to $u_{i,3}$. Further, for each $j \in \{1, \ldots, \mu\}$, there exist arcs from $v_{j,2}$ to t_2 and from $v_{j,4}$ to t_2. Finally, for each $j \in \{1, \ldots, \mu\}$, there exists an arc from s_3 to $v_{j,1}$ and an arc from $v_{j,4}$ to t_3.

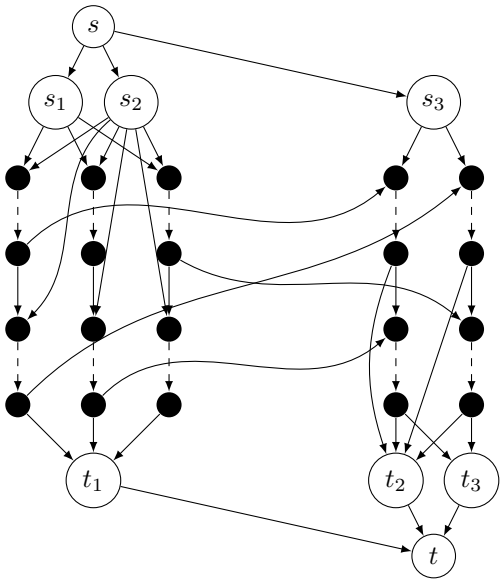

Figure 1 The constructed graph for $X = \{x_1, x_2, x_3\}$ and $C = \{x_1 \vee \overline{x}_2, \overline{x}_1 \vee x_3\}$. The three variable gadgets for x_1, x_2, and x_3 from left to right are shown on the left and the clause gadgets for c_1 and c_2 from left to right are shown on the right. Only the dashed arcs can be interdicted.

We finish the construction by adding the nodes s and t to the graph. For each $k \in \{1, 2, 3\}$, there exists an arc from s to s_k with start time $1 - k$ and traversal time $k - 1$, and an arc from t_k to t with traversal time $3 - k$ (but start time 0).

The budget is chosen to be $B \coloneqq N \cdot \zeta + 2 \cdot \mu - \delta$, which completes the construction of the problem instance.

We show strong $\mathcal{NP}$-hardness by proving that there exists a truth assignment for the variables that satisfies at least δ clauses in the instance of MAX2SAT if and only if there exists a solution for the constructed instance with objective value at least 3. To this end, the following auxiliary result is required.

▶ **Lemma 11.** *Let S be a solution of the constructed instance. The objective value of S is larger than or equal to 3 if and only if the pairs $(s_1, t_1), (s_2, t_2)$, and (s_3, t_3) are separated by S.*

Proof. If one of the pairs $(s_1, t_1), (s_2, t_2)$, or (s_3, t_3) is not separated by S, it follows immediately that, after interdiction, there exists an s-t-path with duration 2. Hence, the solution S has objective value at most 2. To show the other direction, assume that the objective value of the solution is strictly less than 3 and let P_{SD} be a shortest duration path in G_S. If P_{SD} visits both s_k and t_k for some $k \in \{1, 2, 3\}$, we are done. If this is not the case, there are three possible pairs of nodes, one of which must be visited by P_{SD} since its duration is strictly less than 3 and every temporal s-t-path in G_S must visit one of the s_k and one of the t_k.

Case 1: P_{SD} visits s_1 and t_2

This means there exists a subpath P of P_{SD} from s_1 to t_2. The first arc in P leads into one of the nodes $u_{i,1}$ for some $i \in \{1, \ldots, \zeta\}$. By replacing the first arc in P with the arc starting in s_2 and ending in $u_{i,1}$, we obtain a path from s_2 to t_2.

Case 2: P_{SD} visits s_2 and t_3

This means there exists a subpath P of P_{SD} from s_2 to t_3. The last arc in P starts from one of the nodes $v_{j,4}$ for some $j \in \{1, \ldots, \mu\}$. By replacing the last arc in P with the arc starting in $v_{j,4}$ and ending in t_2, we again obtain a path from s_2 to t_2.

Case 3: P_{SD} visits s_1 and t_3

The first and the last arc in the subpath of P_{SD} from s_1 to t_3 can be replaced as in the previous two cases, which again yields a path from s_2 to t_2. ◀

Lemma 11 allows proving strong $\mathcal{NP}$-hardness of T-SDP-IP and T-STP-IP.

▶ **Theorem 12.** *T-SDP-IP and T-STP-IP are strongly $\mathcal{NP}$-hard even on acyclic graphs.*

Proof. We show that there exists a truth assignment for the variables that satisfies at least δ clauses in the instance of MAX2SAT if and only if there exists a solution for the constructed T-SDP-IP instance with objective value at least 3.

First, let x be a truth assignment that satisfies at least δ clauses. We construct an interdiction strategy for the instance of T-SDP-IP with objective value at least 3 as follows. For each $i \in \{1, \dots, \zeta\}$, we interdict the arc from $u_{i,1}$ to $u_{i,2}$ if x_i is true and the arc from $u_{i,3}$ to $u_{i,4}$ if x_i is false. For each $j \in \{1, \dots, \mu\}$, we interdict the arc from $v_{j,1}$ to $v_{j,2}$ if the second literal in clause c_j is fulfilled, the arc from $v_{j,3}$ to $v_{j,4}$ if the second literal of c_j is not fulfilled, but the first is, and both of these arcs if none of the literals are fulfilled. This yields an interdiction strategy S that interdicts ζ arcs of cost N and at most $2 \cdot \mu - \delta$ arcs of cost 1 and, hence, does not exceed the budget.

Due to Lemma 11, it remains to show that the pair (s_k, t_k) is separated by S for each $k \in \{1, 2, 3\}$. Any path from s_1 to t_1 has trace $(s_1, u_{i,1}, u_{i,2}, u_{i,3}, u_{i,4}, t_1)$ for some $i \in \{1, \dots, \zeta\}$. As either the arc from $u_{i,1}$ to $u_{i,2}$ or the arc from $u_{i,3}$ to $u_{i,4}$ is interdicted, the pair (s_1, t_1) is separated by S. Moreover, the analogous argument applied to the clause gadgets shows that the pair (s_3, t_3) is separated by S.

To show that the pair (s_2, t_2) is separated by S, note that each path from s_2 to t_2 contains a subpath with trace $(u_{i,a}, u_{i,a+1}, v_{j,b}, v_{j,b+1})$ where $i \in \{1, \dots, \zeta\}, j \in \{1, \dots, \mu\}$, and $a, b \in \{1, 3\}$. We interdict either the arc from $u_{i,a}$ to $u_{i,a+1}$ if the $(^{b+1}/_2)$-th literal of clause c_j is fulfilled or the arc from $v_{j,b}$ to $v_{j,b+1}$ if it is not. Hence, the pair (s_2, t_2) is separated by S and the objective value of S is at least 3 due to Lemma 11.

For the inverse direction, let $S \subseteq R$ be a removal strategy with $c(S) \leq B = N \cdot \zeta + 2 \cdot \mu - \delta$ and objective value at least 3. In particular, this means that, for each $k \in \{1, 2, 3\}$, the pair (s_k, t_k) is separated by S due to Lemma 11.

In order to separate the pair (s_1, t_1), one arc has to be removed per variable gadget. If more than one arc is removed in any variable gadget, the total removal cost of interdicted arcs in the variable gadgets is at least $N \cdot \zeta + N = N \cdot \zeta + \mu + 1$, which leaves only a budget of $\mu - \delta - 1 < \mu$ for interdicting arcs in the clause gadgets. Hence, there must exist at least one clause gadget in which none of the arcs are interdicted. This implies that the pair (s_3, t_3) is not separated by S, which is a contradiction. Overall, this means that, for each $i \in \{1, \dots, \zeta\}$, either the arc from $u_{i,1}$ to $u_{i,2}$ is interdicted, in which case we set x_i to true, or the arc from $u_{i,3}$ to $u_{i,4}$ is interdicted, in which case we set x_i to false.

It remains to show that the resulting truth assignment fulfills at least δ clauses. As interdicting one arc per variable gadget already costs $N \cdot \zeta$, there is a budget of $2 \cdot \mu - \delta$ left for interdicting arcs in the clause gadgets. In order to separate the pair (s_3, t_3), at least one of the two removable arcs must be removed in each clause gadget. Hence, there are at least δ clause gadgets in which only one of the arcs is removed. We finish the proof by showing that x fulfills all the corresponding clauses c_j.

To this end, we first assume that the arc from $v_{j,3}$ to $v_{j,4}$ is interdicted. If the first literal in c_j is x_i, then there exists a path in G with trace $(s_2, u_{i,1}, u_{i,2}, v_{j,1}, v_{j,2}, t_2)$. As the arc from $v_{j,1}$ to $v_{j,2}$ is not interdicted and the pair (s_2, t_2) must be separated by S, this means that the arc from $u_{i,1}$ to $u_{i,2}$ must be interdicted and, hence, that x_i is set to true, which

shows that x fulfills c_j. If the first literal in c_j is $\overline{x}_i$, then the same arguments hold for the path in G with trace $(s_2, u_{i,3}, u_{i,4}, v_{j,1}, v_{j,2}, t_2)$. The proof for the case when the arc from $v_{j,1}$ to $v_{j,2}$ is interdicted is along the same lines. Hence, at least δ clauses are fulfilled by x, which completes the proof. ◀

Since any solution of the constructed T-SDP-IP instance that does *not* have objective value at least 3 has objective value at most 2, the proof of Theorem 12 further implies the following inapproximability result.

▶ **Corollary 13.** *Unless $\mathcal{P} = \mathcal{NP}$, there exists no polynomial-time approximation algorithm with approximation ratio smaller than $3/2$ for T-SDP-IP or T-STP-IP, even on acyclic graphs.*

In the case of T-SDP-IP, the constructed instance in the reduction can easily be adjusted such that all traversal times are zero. To do so, all traversal times of the outgoing arcs of s are set to 0 and, for each incoming arc of t, the start time is increased by its traversal time and the traversal time is then set to 0. Hence, the results on T-SDP-IP from Theorem 12 and Corollary 13 are also valid for the definition of temporal graphs used in [27].

In the case of T-STP-IP, however, using nonzero traversal times within the reduction is necessary. Indeed, the results on T-STP-IP from Theorem 12 and Corollary 13 do *not* hold for the definition in [27] (unless $\mathcal{P} = \mathcal{NP}$) since T-STP-IP is solvable in polynomial time if all traversal times are zero as it then reduces to the question whether s can be separated from t by an interdiction strategy. It is, however, questionable, whether T-STP-IP has a meaningful interpretation in this case.

We continue by showing that the results of Theorem 12 and Corollary 13 (with a slight modification of the approximation ratio) also hold for instances with unit removal costs and strictly positive traversal times.

The restriction to unit removal costs can be achieved by replacing each arc r in the constructed graph by $c(r)$ identical copies with unit removal cost. Any interdiction strategy can then be assumed to either remove all of these identical copies or none of them. Moreover, since all removal costs have been polynomial in the numbers of variables and clauses of the given MAX2SAT instance, the constructed instance with unit removal costs is still of polynomial size, so the arguments in the proof carry over to this instance.

For the restriction to strictly positive traversal times, note that the constructed graph G is acyclic. In particular, the graph $G - \{s, t\}$ is acyclic. Let $\sigma : V \to \{1, \dots, n-2\}$ be a topological sorting of the nodes in $G - \{s, t\}$, i.e., for each arc r, it holds that $\sigma(\alpha(r)) < \sigma(\omega(r))$. This topological sorting is used to slightly modify the start and traversal times of the arcs in $G - \{s, t\}$. Formally, a function $\bar{\sigma} : V \to \{1, \dots, n-2\}$ is constructed from the topological sorting by setting $\bar{\sigma}(v) := \sigma(v)$ if $v \notin \{s_1, s_2, s_3, t_1, t_2, t_3\}$, and $\bar{\sigma}(s_i) := 1$ and $\bar{\sigma}(t_i) := n - 2$ for each $i \in \{1, 2, 3\}$. We then redefine the start and traversal times in G. To this end, let $\varepsilon \in (0, 1)$. For each arc r that is not incident to s or t, we set the start time to $-\varepsilon/2 \cdot (n-3) \cdot (n - 2 - \bar{\sigma}(\alpha(r)))$ and the traversal time to $\varepsilon/2 \cdot (n-3) \cdot (\bar{\sigma}(\omega(r)) - \bar{\sigma}(\alpha(r)))$, which means that it arrives in $\omega(r)$ at time $-\varepsilon/2 \cdot (n-3) \cdot (n - 2 - \bar{\sigma}(\omega(r)))$. We further set the start time of the arc from s to s_1 to $-\varepsilon$ and its traversal time to $\varepsilon/2$, and we decrease the traversal times of the other two outgoing arcs of s by $\varepsilon/2$. Hence, all outgoing arcs of s have arrival time $-\varepsilon/2$. Moreover, we set the traversal time of the arc from t_3 to t to ε.

The proof of Theorem 12 for this new instance is along the same lines as before and the statement of Corollary 13 must be slightly changed (see Corollary 14). Note that the graph with the updated start and traversal times does not admit waiting in any node except for s and t as, for any node $v \in V \setminus \{s, t\}$, all incoming arcs arrive and all outgoing arcs start at time $-\varepsilon/2 \cdot (n-3) \cdot (n - 2 - \bar{\sigma}(v))$. The result, hence, holds for both problems T-SDP-IP and T-STP-IP.

Hence, when strictly positive traversal times on all arcs are additionally assumed, Theorem 12 still holds, but Corollary 13 has to be adapted as follows:

▶ **Corollary 14.** *Unless $\mathcal{P} = \mathcal{NP}$, there exists no polynomial-time approximation algorithm with approximation ratio smaller than $(3/2+\varepsilon)$ for T-SDP-IP and T-STP-IP on acyclic graphs with positive traversal times for any $\varepsilon > 0$.*

3.3.1 Polynomial-Time Solvability on Extension-Parallel Graphs

In this section, we show that T-SDP-IP, T-STP-IP, and the static version S-SP-IP are polynomial-time solvable on extension-parallel (temporal) graphs.

A temporal graph consisting of two nodes s and t, and a single temporal arc from s to t is called a *temporal one-arc graph*. A temporal graph with two distinguished vertices s (the source) and t (the sink) is *series-parallel* if it is obtained from a set of temporal one-arc graphs by a finite sequence of series compositions (identifying the sink of the first graph with the source of the second graph) and parallel compositions (identifying the sources of the two graphs and identifying the sinks of the two graphs). If, further, for every series composition, one of the two composed graphs is a temporal one-arc graph, the graph is called *extension-parallel*. The definitions of series- and extension-parallel *static* graphs is completely analogous and can be found, e.g., in [19].

The *decomposition tree* T_G of a series-parallel (temporal) graph G is a binary tree, where the leaves represent the arcs in the graph and the inner nodes labeled by S (series composition) or P (parallel composition) represent the types of compositions used to construct the graph. The decomposition tree can be computed in linear time [36] and it can easily be seen that a series-parallel (temporal) graph is extension-parallel if and only if every inner node of T_G that is labeled by S has one child that is a leaf of T_G.
The following property of extension-parallel static graphs is used in our algorithm.

▶ **Lemma 15.** *Let $G = (V, R)$ be an extension-parallel static graph. Then there exists a subset $\bar{R} \subseteq R$ of arcs such that*
1. *each s-t-path in G contains exactly one arc from $\bar{R}$, and*
2. *each arc $r \in \bar{R}$ is contained in exactly one s-t-path P_r in G.*

Proof. We present an algorithm that constructs $\bar{R}$ and a corresponding s-t-path P_r for each $r \in \bar{R}$. Note that, since the graph is acyclic, the path P_r is uniquely determined by the set of arcs it traverses. Hence, we slightly abuse notation and identify each path P_r with the corresponding set of arcs. The idea of the algorithm is to process the nodes in the decomposition tree starting at the leaves by iteratively joining two already processed components of the graph until we reach the root node and obtain the final set $\bar{R}$.

Initially, we set $\bar{R} := R$ and $P_r := \{r\}$ for each $r \in R$ and mark all leaf nodes in the decomposition tree as processed. While not all nodes in the decomposition tree are marked as processed, we take an unprocessed (inner) node v in the decomposition tree whose two children have both been processed. If v is labeled by P, we simply mark v as processed while changing neither the set $\bar{R}$ nor the paths P_r, $r \in \bar{R}$. If v is labeled by S, at least one of its children must be a leaf corresponding to an arc r. If only one of the children is a leaf node, then we remove r from $\bar{R}$ and delete P_r. Further, we add r to all paths $P_{r'}$ for which the leaf node that corresponds to r' is a successor of the non-leaf child of v in the decomposition tree. If both children of v are leaves, then the arc that corresponds to its right child is removed from $\bar{R}$ and added to the path $P_{r'}$, where r' is the arc that corresponds to the left child of v. We then mark v as processed and proceed.

To show the correctness of the algorithm, note that each node v in the decomposition tree can be associated with the subgraph G_v of G whose arc set consists of those arcs that correspond to leaf nodes in the decomposition tree that are successors of v.

We claim that, after each iteration, for each processed node v that either has no parent (i.e, v is the root node) or whose parent is still unprocessed, it holds that $\bar{R}$ restricted to the arc set of G_v fulfills the properties from the lemma for G_v.

This is clearly the case when only the leaves have been processed since every subgraph G_v is then a one-arc graph. Now assume that the claim holds at the beginning of an iteration and let v be the node in the decomposition tree that is processed in the iteration. If v is labeled by P, each s-t-path in G_v is either completely contained in the graph associated with the left child of v in the decomposition tree or in the graph associated with the right child. Hence, since $\bar{R}$ and the paths P_r, $r \in \bar{R}$, are left unchanged, the claim also holds after processing v. If the processed node v is labeled by S and both its children are leaves, the claim clearly remains true. If the processed node v is labeled by S and only one of its children is a leaf, then this series composition corresponds to prepending or appending an additional arc to the graph G_w, where w denotes the non-leaf child of v. Hence, after the series composition, each path in G_w is extended by the arc r that corresponds to the leaf child of v, which is precisely what the algorithm does. Moreover, r is removed from $\bar{R}$ and P_r is deleted, which ensures that uniqueness is preserved in both properties from the lemma. Hence, the claim also holds after the iteration, which completes the proof. ◀

Note that the proof of Lemma 15 is constructive and the set $\bar{R}$ together with the paths P_r for $r \in \bar{R}$ can be obtained in $\mathcal{O}(m^2)$ time. In the following, given an extension-parallel temporal graph, we let $\bar{R}$ denote a subset of arcs that satisfies the properties of Lemma 15 in the underlying static graph. For each arc $r \in \bar{R}$, we remove r from the graph and from $\bar{R}$ if the corresponding s-t-path P_r in the underlying static graph is not a (temporal) s-t-path in the temporal graph. Note that this does not destroy any temporal s-t-paths.

The idea of the polynomial-time algorithm to solve T-SDP-IP, T-STP-IP, and the static version S-SP-IP is similar to the idea of the algorithm for T-LSP-IP from Section 3.1. For ease of notation, the following exposition is restricted to T-SDP-IP. It is discussed later how the arguments can be modified for the other two problems.

Let $\text{dura}_1 < \text{dura}_2 < \cdots < \text{dura}_l$ denote the distinct durations of s-t-paths in G sorted in increasing order. Further, for $k \in \{1, \dots, l\}$, we define $G^{\text{SD},k}$ as the temporal graph that results from G by removing each arc $r \in \bar{R}$ for which P_r has duration at least dura_k. For completeness, we also define $G^{\text{SD},l+1} := G$. The following proposition and its proof are similar to Proposition 4 and the corresponding proof.

▶ **Proposition 16.** *Let $k \in \{1, \dots, l\}$. There exists an interdiction strategy S^k that separates s from t in $G^{\text{SD},k}$ if and only if there exists an interdiction strategy S in G with objective value at least dura_k.*

Proof. Let S^k be an interdiction strategy that separates s from t in $G^{\text{SD},k}$. Then, after interdicting the same set $S := S^k$ of arcs in G, no s-t-path P in G_S can have duration less than dura_k (otherwise, P would also be an s-t-path in $G^{\text{SD},k}_{S^k}$ as (1) no arc in P is in $S = S^k$, and (2) the unique arc r in P contained in $\bar{R}$ satisfies $P_r = P$ and, thus, $\text{dura}(P_r) = \text{dura}(P) < \text{dura}_k$). Hence, all s-t-paths in G_S have duration at least dura_k, i.e., S has objective value at least dura_k.

Conversely, let S be an interdiction strategy in G with objective value at least dura_k. Then, no s-t-path in G_S can have duration less than dura_k, so the interdiction strategy $S^k := S \setminus \{r \in \bar{R} \,|\, \text{dura}(P_r) \geq \text{dura}_k\}$ separates s from t in $G^{\text{SD},k}$. ◀

As in the algorithm presented in Section 3.1, the idea of the algorithm is to use binary search in order to find $k^\star \in \{1, \dots, l\}$ such that s can be separated from t in $G^{\mathrm{SD},k^\star}$, but s cannot be separated from t in $G^{\mathrm{SD},k^\star+1}$. Such a $k^\star$ exists whenever s cannot already be separated from t in the whole graph $G = G^{\mathrm{SD},l+1}$, i.e., whenever the optimal objective value is not equal to $+\infty$.

As shown in Section 3.1, deciding whether a node s and can be separated from another node t with a given interdiction budget in an arbitrary temporal graph is possible in polynomial time. Further, Lemma 15 implies that the total number of s-t-paths is bounded by the number of arcs in the graph. Consequently, the number l of distinct durations of s-t-paths is polynomial in the input size. Altogether, the proposed algorithm runs in polynomial time.

To extend the result to the problems T-STP-IP and S-SP-IP, one observes the distinct traversal times or lengths of s-t-paths, respectively, to construct the subgraphs used in the algorithm. All arguments then work along the same lines.
The following theorem summarizes the main results of this section.

▶ **Theorem 17.** *There exist polynomial-time algorithms for T-SDP-IP, T-STP-IP, and S-SP-IP on extension-parallel (temporal) graphs with running time in* $\mathcal{O}(m^2 + \log(m) \cdot T_{\mathrm{MC}}(n \cdot m, n \cdot m))$ *for the temporal versions and running time in* $\mathcal{O}(m^2 + \log(m) \cdot T_{\mathrm{MC}}(n, m))$ *for the static version, where* $T_{\mathrm{MC}}(\bar{n}, \bar{m})$ *is the time required to compute a minimum s-t-cut in a static graph with* $\bar{n}$ *nodes and* $\bar{m}$ *arcs.*

4 Extensions

In this section, we study two extensions of the temporal shortest path interdiction problem. The first extension is motivated by [13] and allows for continuous-time availability of arcs. It is shown that even a slight generalization makes it hard to decide whether the nodes s and t can be separated by an interdiction strategy in a temporal graph. The second extension, motivated by [14], imposes an additional constraint on the maximum waiting time in a node. It is shown that the additional constraint does not change the results from Section 3.

4.1 Continuous Time Availability of Arcs

In this section, a slightly more general model of temporal graphs is investigated, where the start time $\tau(r)$ of a temporal arc r is not given by one fixed point in time, but rather by a closed interval $[\tau^{\mathrm{l}}(r), \tau^{\mathrm{u}}(r)] =: \tau(r)$. The arc can then be entered at any time $\tau \in \tau(r)$, leading to an arrival time of $\tau + \lambda(r)$ at $\omega(r)$. In the following, we therefore no longer speak of a start time, but of an *availability interval*. The resulting temporal graphs where arcs are available during availability intervals are called *continuous-time temporal graphs*. Note that the class of continuous-time temporal graphs comprises the class of discrete-time temporal graphs, which correspond to continuous-time temporal graphs in which each availability interval only consists of a single point.

While a temporal path in a discrete-time temporal graph is given by a sequence of temporal arcs, the definition has to be slightly adapted in the continuous-time case. A *(continuous-time) temporal path* in a continuous-time temporal graph is a sequence $P = ((r_1, \tau_1), \dots, (r_k, \tau_k))$ of pairs of an arc and a start time with $\tau_i \in \tau(r_i)$ for each $i \in \{1, \dots, k\}$ such that $\omega(r_i) = \alpha(r_{i+1})$ and $\tau_i + \lambda(r_i) \leq \tau_{i+1}$ for each $i \in \{1, \dots, k-1\}$.

We show that, given a continuous-time temporal graph G and a positive integer B, it is strongly $\mathcal{NP}$-hard to decide whether a pair (s, t) of nodes can be separated by removing at most B arcs from G. This immediately implies that all variants of temporal shortest

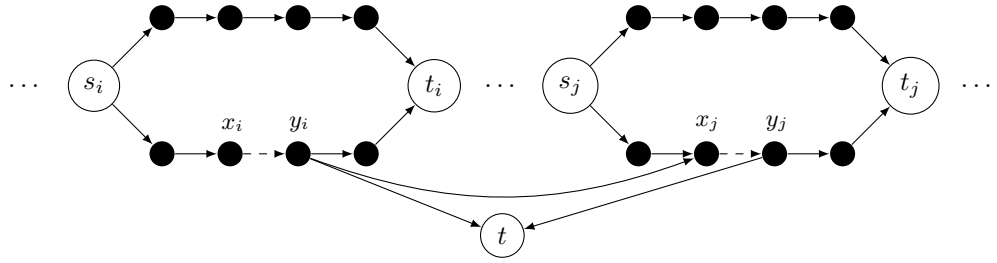

Figure 2 The gadgets in G for two nodes i and j that are adjacent in G' (where $i < j$). Only the dashed arcs can be interdicted.

path interdiction problems studied in this paper are strongly $\mathcal{NP}$-hard on continuous-time temporal graphs, and that they do not not even admit polynomial-time approximation algorithms with a bounded approximation ratio (unless $\mathcal{P} = \mathcal{NP}$).

The reduction, which is similar to the one presented in [3], is from the well-known (strongly) $\mathcal{NP}$-hard node cover problem, which is defined as follows:

INSTANCE:	An undirected (static) graph $G' = (V', E)$ and a positive integer $B' \leq \lvert V' \rvert$
QUESTION:	Is there a subset $\bar{V} \subseteq V'$ of nodes with $\lvert \bar{V} \rvert \leq B'$ such that each edge in E is incident to at least one node in $\bar{V}$?

Given an instance of node cover, a continuous-time temporal graph G and a budget B are constructed as follows. The budget is chosen as $B := B'$. For the construction of the continuous-time temporal graph G, it is assumed without loss of generality that $V' = \{1, \dots, n'\}$. For each node $i \in \{1, \dots, n'\}$, a gadget consisting of two parallel paths of length five is constructed. These paths are referred to as the *upper* and *lower* path of the gadget. The start node of the two paths is referred to as s_i and the end node of the two paths as t_i. Further, the start and end node of the third arc in the lower path are referred to as x_i and y_i, respectively. All arcs in a gadget, except for the third arc in the lower path, have removal cost $B+1$ and availability interval $[0, 5n']$. The third arc in the lower path has removal cost 1 and availability interval $[5i-4, 5i-3]$, and we identify removing this arc with including node i in the node cover. All arcs in the gadget have traversal time 1.

The gadgets are connected by identifying t_i with s_{i+1} for all $i \in \{1, \dots, n'-1\}$. Further, for an edge $e \in E$ that is incident to the nodes i and j with $i < j$, there exists an arc from y_i to x_j with availability interval $[5i-2, 5i-2]$, traversal time $5(j-i)-2$, and removal cost $B+1$. This arc is called the *shortcut* from i to j.

The node s for the instance of temporal shortest path interdiction is s_1. The node t is added and, for each $i \in \{1, \dots, n'\}$, there exists an arc from y_i to t with availability interval $[5i-3, 5i-3]$, traversal time 0, and removal cost $B+1$. An illustration of the construction is provided in Figure 2.

To achieve unit removal costs, we can simply replace each arc r by $c(r)$ many identical copies with unit removal cost (as for the proof of Theorem 12 and Corollary 13 for unit removal costs). Note that this conserves the polynomial size of the constructed instance. Using this construction, we prove the following theorem.

▶ **Theorem 18.** *Deciding whether a pair (s,t) of nodes can be separated by removing at most B arcs from a continuous-time temporal graph is strongly $\mathcal{NP}$-complete.*

Proof. The problem is clearly in $\mathcal{NP}$ since it can be easily checked in polynomial time whether a given set of at most B arcs separates s from t. To show $\mathcal{NP}$-completeness, let S be an interdiction strategy for the constructed instance. Observe that each s-t-path in G_S

traverses at least one shortcut from i to j for some $i, j \in \{1, \dots, n'\}$. This is possible if and only if none of the removable arcs in gadgets i and j are removed by the interdiction strategy. By identifying the interdiction of an interdictable arc in a gadget i with the inclusion of node i in the node cover, it follows that there exists a node cover of size B' $(= B)$ if and only if there exists an interdiction strategy in G that removes at most B arcs. ◀

4.2 Waiting Time Constraints

The definition of an s-t-path provided in Section 2 allows to wait at nodes for any length of time. However, arbitrarily long waiting times are often undesired in real-world problems such as, e.g., packet routing in communication networks. To this end, the problem of finding a Δ-*restless temporal s-t-path* that cannot wait longer than a given amount of time Δ in any node except s and t has been defined (see [14]). As shown in [14], deciding whether a *simple* Δ-restless s-t-path (i.e., a Δ-restless s-t-path that does not visit any node more than once) exists is strongly $\mathcal{NP}$-hard for any $\Delta \geq 0$.[3] However, in the setting considered here where paths are not required to be simple, this problem is polynomial-time solvable. A Dijkstra-like polynomial-time algorithm for computing *not necessarily simple* restless paths in temporal graphs is presented in [5].

In this section, we show how the time-expanded graph introduced in Section 3.1 can be modified to account for additional waiting time constraints. Further, we show that the complexity of the four versions of temporal shortest path interdiction does not change under additional waiting time constraints.

Within this section, we assume that s has no incoming arcs and t has no outgoing arcs. This assumption does not impose a loss of generality since a shortest s-t-path (with respect to any of the definitions of "shortest") that uses such an arc could be transformed into one that does not.

For the construction of the time-expanded graph $H^{\text{te}} = (V^{\text{te}}, R^{\text{te}})$ under waiting time constraints, recall the set $T = \{\phi_1, \dots, \phi_j\}$ of crucial times, which are indexed in increasing order. Similar to the construction of the time-expanded graph in Section 3.1, we introduce a node (v, ϕ_i) for every $v \in V$ and $i \in \{1, \dots, j\}$. For $v \in \{s, t\}$ and $i \in \{1, \dots, j-1\}$, there exists an arc from (v, ϕ_i) to (v, ϕ_{i+1}), which represents waiting at s before the start of the path or waiting at t after having arrived. For each arc $r \in R$, an additional node u_r is introduced. Further, there exists an arc in R^{te} from $(\alpha(r), \tau(r))$ to u_r and an arc from u_r to any node $(\omega(r), \phi)$ with $\phi \in [\tau(r) + \lambda(r), \tau(r) + \lambda(r) + \Delta]$. Traversing the arc from $(\alpha(r), \tau(r))$ to u_r and then the arc from u_r to some node $(\omega(r), \phi)$ represents traversing r in the temporal graph and entering the next arc in the path (if $\omega(r) \neq t$) exactly at time ϕ. This completes the construction of H^{te}.

To show a one to one correspondence between s-t-paths in G and (s, ϕ_1)-(t, ϕ_j)-paths in H^{te}, note that there are no parallel arcs in H^{te}, which implies that any path in H^{te} is uniquely given by its trace.

▶ **Observation 19.** *There exists a Δ-restless s-t-path in G if and only if there exists a (s, ϕ_1)-(t, ϕ_j)-path in H^{te}.*

[3] It is worth noting that the definition of temporal graphs in [14] is slightly different. They state that the problem is strongly $\mathcal{NP}$-hard for any $\Delta \geq 1$, but, indeed, the same proof of hardness with a slight modification holds true for the definition of temporal graphs used here for any $\Delta \geq 0$.

Proof. Let $P = (r_1, \dots, r_k)$ be a Δ-restless s-t-path in G. Then we claim that the unique path with trace

$$((s, \phi_1), \dots, (s, \tau(r_1)), u_{r_1}, (\omega(r_1), \tau(r_2)), u_{r_2}, (\omega(r_2), \tau(r_3)), \dots, (t, \tau(r_k) + \lambda(r_k)), \dots, (t, \phi_j))$$

is a (s, ϕ_1)-(t, ϕ_j) path in H^{te}. Since, for $v \in \{s, t\}$ and $i \in \{1, \dots, j-1\}$, there exists an arc from (v, ϕ_i) to (v, ϕ_{i+1}), the arcs from (s, ϕ_1) to $(s, \tau(r_1))$ and the arcs from $(t, \tau(r_k) + \lambda(r_k))$ to (t, ϕ_j) are in H^{te}. Moreover, the arc from $(\alpha(r_i), \tau(r_i))$ to u_{r_i} is in H^{te} for $i \in \{1, \dots, k\}$, and the arc from u_{r_i} to $(\omega(r_i), \tau(r_{i+1}))$ is in H^{te} for $i \in \{1, \dots, k-1\}$ since P is Δ-restless.

Conversely let P' be an (s, ϕ_1)-(t, ϕ_j)-path in H^{te}. From the construction, it immediately follows that the trace of P' must be of the above form and, by the same arguments as above, it follows that $(r_1, \dots, r_k)$ is a Δ-restless s-t-path in G. ◀

To show that T-LSP-IP and T-EAP-IP remain polynomial-time solvable, we assign removal costs to $H^{\text{te}} = (V^{\text{te}}, R^{\text{te}})$. For each arc $r \in R$, the (unique) incoming arc of u_r has removal cost $c(r)$. All other arcs have removal cost $B + 1$. With this construction, we observe the following.

▶ **Observation 20.** *There exists an interdiction strategy S such that there does not exist a Δ-restless path in G_S if and only if there exists an interdiction strategy S' separating s^{te} from t^{te} in H^{te}.*

Using the algorithm proposed in Sections 3.1 and 3.2 together with the time-expanded graph $H^{\text{te}} = (V^{\text{te}}, R^{\text{te}})$ under waiting time constraints constructed in this section, it follows that the problems remain polynomial-time solvable under waiting time constraints.

▶ **Theorem 21.** *There exists a polynomial-time algorithm for solving T-LSP-IP and T-EAP-IP under waiting time constraints for each $\Delta \geq 0$.*

We proceed with assessing the complexity of T-SDP-IP and T-STP-IP under waiting time constraints. When taking a closer look at the reduction provided in Section 3.3, every s-t-path in the constructed instance is 0-restless. This immediately implies that T-SDP-IP under waiting time constraints is strongly $\mathcal{NP}$-hard for every $\Delta \geq 0$. Further, on instances where waiting in any node except s and t is impossible, the problems T-SDP-IP and T-STP-IP are equivalent. This yields the following theorem.

▶ **Theorem 22.** *The problems T-SDP-IP and T-STP-IP are strongly $\mathcal{NP}$-hard under waiting time constraints for each $\Delta \geq 0$.*

To close this chapter, we argue that T-SDP-IP and T-STP-IP are still polynomial-time solvable on extension-parallel temporal graphs under waiting time constraints. To this end, when removing each arc r whose corresponding s-t-path P_r is not a temporal path from the graph and from the set $\bar{R}$ as in Lemma 15, we additionally check whether P_r is Δ-restless and remove r if this is not the case. Afterwards, the graph contains exactly the Δ-restless temporal s-t-paths and the algorithm presented in Section 3.3.1 can be used to solve T-SDP-IP and T-STP-IP on extension-parallel temporal graphs under waiting time constraints, which yields the following theorem.

▶ **Theorem 23.** *There exists a polynomial-time algorithm for T-SDP-IP and T-STP-IP on extension-parallel (temporal) graphs under waiting time constraints for each $\Delta \geq 0$.*

It is worth noting that all results presented in this section can easily be adapted to the general case where waiting times are constrained node-wise instead of globally. The only difference to the global case is in the construction of the time-expanded graph, where the node-wise constriction is enforced by the outgoing arcs of the nodes u_r for $r \in R$.

5 Conclusion

In this paper, the complexity of four different versions of temporal shortest path interdiction is analyzed. While the latest start and the earliest arrival path interdiction problems are shown to be solvable in polynomial time, the shortest duration and the shortest traversal path interdiction problems are strongly $\mathcal{NP}$-hard. It is particularly interesting that, even though *temporal* shortest path interdiction seems more complex than its static counterpart, which is known to be strongly $\mathcal{NP}$-hard, there are versions of temporal shortest path interdiction problems that are polynomially solvable. We further provide polynomial-time algorithms for the two hard problems on extension-parallel temporal graphs, which can also be transferred to the static shortest path interdiction problem.

An interesting direction for future work could be to study temporal shortest path interdiction problems for other types of modifications than arc removal. For example, one could consider the problem of worsening (or improving) the latest start time, the earliest arrival time, the shortest duration, or the shortest traversal time as much as possible by changing a given number of start times of arcs in a temporal graph.

References

1 E. C. Akrida, G. B. Mertzios, P. G. Spirakis, and V. Zamaraev. Temporal vertex cover with a sliding time window. *Journal of Computer and System Sciences*, 107:108–123, 2020. doi:10.1016/j.jcss.2019.08.002.

2 M. O. Ball, B. L. Golden, and R. V. Vohra. Finding the most vital arcs in a network. *Operations Research Letters*, 8(2):73–76, 1989. doi:10.1016/0167-6377(89)90003-5.

3 A. Bar-Noy, S. Khuller, and B. Schieber. The complexity of finding most vital arcs and nodes. Technical Report CS-TR-3539, University of Maryland, 1995.

4 B. M. Behring, A. Rizzo, and M. Porfiri. How adherence to public health measures shapes epidemic spreading: A temporal network model. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(4):043115, 2021. doi:10.1063/5.0041993.

5 M. Bentert, A.-S. Himmel, A. Nichterlein, and R. Niedermeier. Efficient computation of optimal temporal walks under waiting-time constraints. *Applied Network Science*, 5(1):73, 2020. doi:10.1007/s41109-020-00311-0.

6 C. Bentz. On the hardness of finding near-optimal multicuts in directed acyclic graphs. *Theoretical Computer Science*, 412(39):5325–5332, 2011. doi:10.1016/j.tcs.2011.06.003.

7 K. A. Berman. Vulnerability of scheduled networks and a generalization of Menger's Theorem. *Networks*, 28(3):125–134, 1996. doi:10.1002/(SICI)1097-0037(199610)28:3<125::AID-NET1>3.0.CO;2-P.

8 J. Boeckmann and C. Thielen. A $(B+1)$-approximation for network flow interdiction with unit costs. *Discrete Applied Mathematics (online first)*, pages 1–14, 2021. doi:10.1016/j.dam.2021.07.008.

9 F. Brunelli, P. Crescenzi, and L. Viennot. On computing Pareto optimal paths in weighted time-dependent networks. *Information Processing Letters*, 168:106086, 2021. doi:10.1016/j.ipl.2020.106086.

10 B. Bui-Xuan, A. Ferreira, and A. Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(2):267–285, 2003. doi:10.1142/S0129054103001728.

11 C. Burch, R. Carr, S. Krumke, M. Marathe, C. Phillips, and E. Sundberg. A decomposition-based pseudoapproximation algorithm for network flow inhibition. In D. L. Woodruff, editor, *Network Interdiction and Stochastic Integer Programming*, chapter 1, pages 51–68. Kluwer Academic Press, 2003. doi:10.1007/0-306-48109-X_3.

12 S. Busam, L. E. Schäfer, and S. Ruzika. The two player shortest path network interdiction problem, 2020. arXiv:2004.08338.

13 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):387–408, 2012. doi:10.1080/17445760.2012.668546.

14 A. Casteigts, A.-S. Himmel, H. Molter, and P. Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021. doi:10.1007/s00453-021-00831-w.

15 S. R. Chestnut and R. Zenklusen. Hardness and approximation for network flow interdiction. *Networks*, 69(4):378–387, 2017. doi:10.1002/net.21739.

16 A. Deligkas and I. Potapov. Optimizing reachability sets in temporal graphs by delaying. *Information and Computation*, 285:104890, 2022. doi:10.1016/j.ic.2022.104890.

17 J. Enright, K. Meeks, G. B. Mertzios, and V. Zamaraev. Deleting edges to restrict the size of an epidemic in temporal networks. *Journal of Computer and System Sciences*, 119:60–77, 2021. doi:10.1016/j.jcss.2021.01.007.

18 J. Enright, K. Meeks, and F. Skerman. Assigning times to minimise reachability in temporal graphs. *Journal of Computer and System Sciences*, 115:169–186, 2021. doi:10.1016/j.jcss.2020.08.001.

19 A. Epstein, M. Feldman, and Y. Mansour. Strong equilibrium in cost sharing connection games. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC)*, pages 84–92, 2007. doi:10.1145/1250910.1250924.

20 T. Fluschnik, H. Molter, R. Niedermeier, M. Renken, and P. Zschoche. Temporal graph classes: A view through temporal separators. *Theoretical Computer Science*, 806:197–218, 2020. doi:10.1016/j.tcs.2019.03.031.

21 T. Holzmann and J.C. Smith. The shortest path interdiction problem with randomized interdiction strategies: Complexity and algorithms. *Operations Research*, 69(1):82–99, 2021. doi:10.1287/opre.2020.2023.

22 A. Ibiapina, R. Lopes, A. Marino, and A. Silva. Menger's Theorem for temporal paths (not walks), 2022. arXiv:2206.15251.

23 D. Kempke, J. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the 32nd ACM Symposium on the Theory of Computing (STOC)*, pages 504–513, 2000.

24 L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. *Theory of Computing Systems*, 43(2):204–233, 2008. doi:10.1007/s00224-007-9025-6.

25 N. Maack, H. Molter, R. Niedermeier, and M. Renken. On finding separators in temporal split and permutation graphs. *Journal of Computer and System Sciences*, 135:1–14, 2023. doi:10.1016/j.jcss.2023.01.004.

26 O. Michail and P.G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical Computer Science*, 634:1–23, 2016. doi:10.1016/j.tcs.2016.04.006.

27 H. Molter. *Classic Graph Problems Made Temporal – A Parameterized Complexity Analysis*. PhD thesis, Technische Universität Berlin, 2020.

28 H. Molter. The complexity of finding temporal separators under waiting time constraints. *Information Processing Letters*, 175:106229, 2022. doi:10.1016/j.ipl.2021.106229.

29 H. Molter, M. Renken, and P. Zschoche. Temporal reachability minimization: Delaying vs. deleting, 2021. arXiv:2102.10814.

30 P. Mutzel and L. Oettershagen. On the enumeration of bicriteria temporal paths. In *Proceedings of the 15th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, volume 11436 of *Lecture Notes in Computer Science*, pages 518–535, 2019. doi:10.1007/978-3-030-14812-6_32.

31 L. Oettershagen. *Temporal Graph Algorithms*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 2022.

32 J. B. Orlin. Minimum convex cost dynamic network flows. *Mathematics of Operations Research*, 9(2):190–207, 1984. doi:10.1287/moor.9.2.190.

33 C. Phillips. The network inhibition problem. In *Proceedings of the 25th ACM Symposium on the Theory of Computing (STOC)*, pages 776–785, 1993.

34 J. A. Sefair and J. C. Smith. Dynamic shortest-path interdiction. *Networks*, 68(4):315–330, 2016. doi:10.1002/net.21712.

35 J.C. Smith and Y. Song. A survey of network interdiction models and algorithms. *European Journal of Operational Research*, 283(3):797–811, 2020. doi:10.1016/j.ejor.2019.06.024.

36 J. Valdes, R. E. Tarjan, and E. L. Lawler. The recognition of series parallel digraphs. *SIAM Journal on Computing*, 11(2):298–313, 1982. doi:10.1145/800135.804393.

37 R. D. Wollmer. Removing arcs from a network. *Operations Research*, 12(6):934–940, 1964. doi:10.1287/opre.12.6.934.

38 H. Wu, J. Cheng, Y. Ke, S. Huang, Y. Huang, and H. Wu. Efficient algorithms for temporal path computation. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2927–2942, 2016. doi:10.1109/TKDE.2016.2594065.

39 P. Zschoche, T. Fluschnik, H. Molter, and R. Niedermeier. The complexity of finding small separators in temporal graphs. *Journal of Computer and System Sciences*, 107:72–92, 2020. doi:10.1016/j.jcss.2019.07.006.