Minimum Temporal Spanners in Happy Graphs

Hendrik Molter

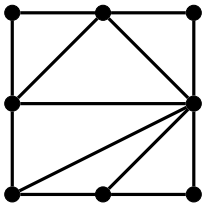
Hasso Plattner Institute, Potsdam, Germany

Algorithmic Aspects of Temporal Graphs IX

Based on joint work with Arnaud Casteigts and Meirav Zehavi.

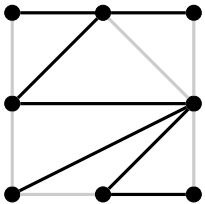
Motivation

Spanning Trees



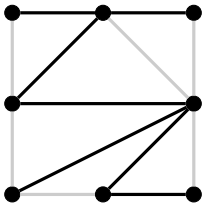
Motivation

Spanning Trees



Motivation

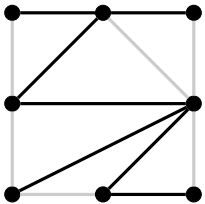
Spanning Trees



Most fundamental primitive in algorithmic graph theory.

Motivation

Spanning Trees

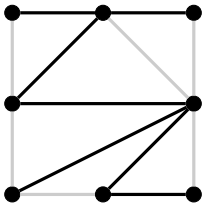


Most fundamental primitive in algorithmic graph theory.

Temporal Spanners

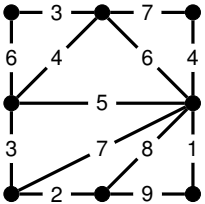
Motivation

Spanning Trees



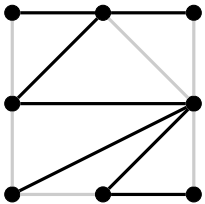
Most fundamental primitive in algorithmic graph theory.

Temporal Spanners



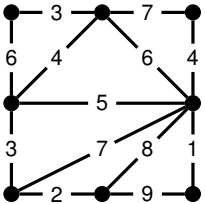
Motivation

Spanning Trees



Most fundamental primitive in algorithmic graph theory.

Temporal Spanners



Canonical generalization to the temporal setting.

Temporal Graphs and Temporal Connectivity: Notation and Definition

Temporal Graph

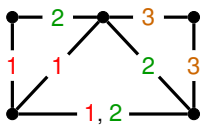
$\mathscr{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathscr{G}$.

Temporal Graphs and Temporal Connectivity: Notation and Definition

Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

$\mathcal{G}$:

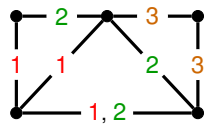


Temporal Graphs and Temporal Connectivity: Notation and Definition

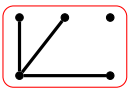
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

$\mathcal{G}$:



Layers:

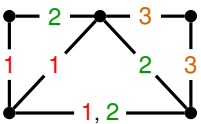


Temporal Graphs and Temporal Connectivity: Notation and Definition

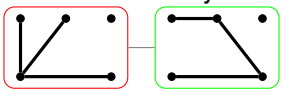
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

$\mathcal{G}$:



Layers:

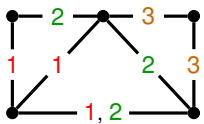


Temporal Graphs and Temporal Connectivity: Notation and Definition

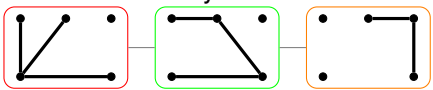
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

$\mathcal{G}$:



Layers:

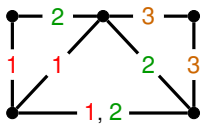


Temporal Graphs and Temporal Connectivity: Notation and Definition

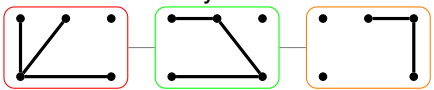
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

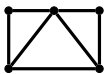
$\mathcal{G}$:



Layers:



Underlying graph:

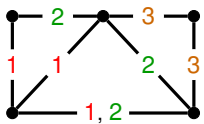


Temporal Graphs and Temporal Connectivity: Notation and Definition

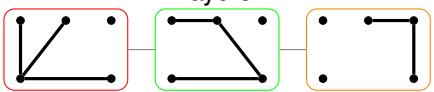
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

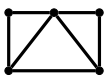
$\mathcal{G}$:



Layers:



Underlying graph:



Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

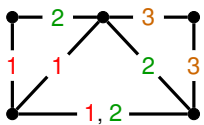
- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).

Temporal Graphs and Temporal Connectivity: Notation and Definition

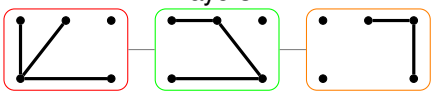
Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.

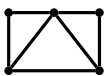
$\mathcal{G}$:



Layers:



Underlying graph:



Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

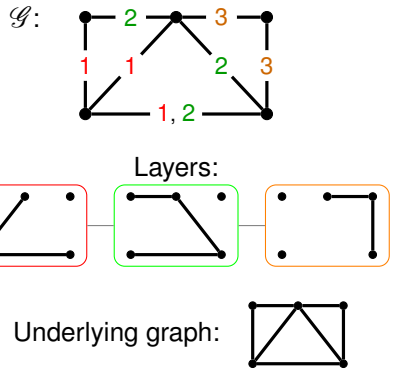
- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).

Not a temporal path.

Temporal Graphs and Temporal Connectivity: Notation and Definition

Temporal Graph

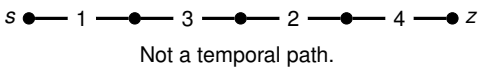
$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.



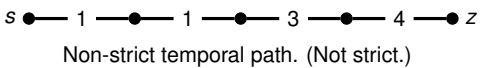
Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).



Not a temporal path.

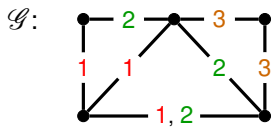


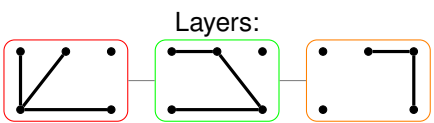
Non-strict temporal path. (Not strict.)

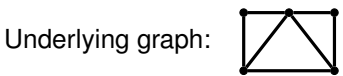
Temporal Graphs and Temporal Connectivity: Notation and Definition

Temporal Graph

$\mathcal{G} = (V, (E_i)_{i \in [T]})$: vertex set V, edge sets $E_1, \ldots, E_T$ over V, where T is lifetime of $\mathcal{G}$.



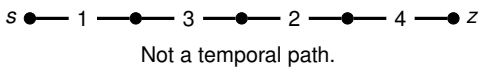




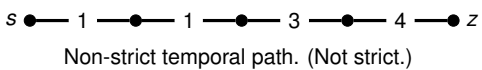
Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

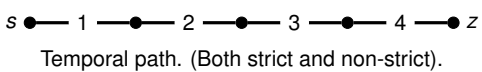
- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).



Not a temporal path.



Non-strict temporal path. (Not strict.)



Temporal path. (Both strict and non-strict).

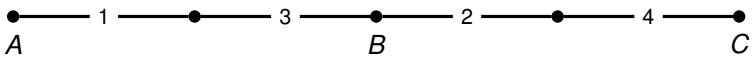
Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

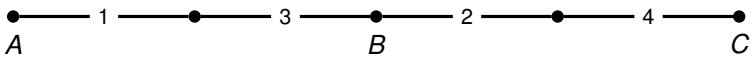
Temporal setting:



Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:

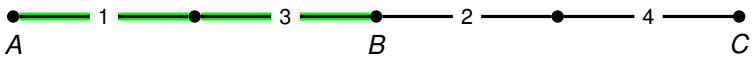


- **Not symmetric!**

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:

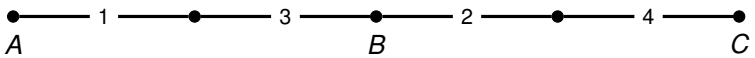


- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:

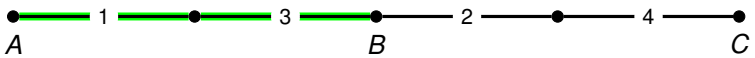


- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.
- **Not transitive!**

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:



- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.
- **Not transitive!**
  There is a temporal path from A to B and from B to C, but not from A to C.

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

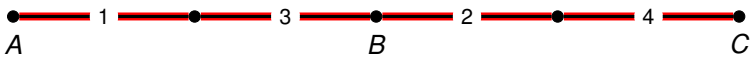
Temporal setting:

- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.
- **Not transitive!**
  There is a temporal path from A to B and from B to C, but not from A to C.

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:

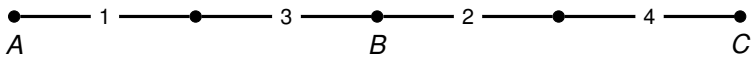


- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.
- **Not transitive!**
  There is a temporal path from A to B and from B to C, but not from A to C.

Oddities of Temporal Connectivity

Connectivity in the static (undirected) setting: symmetric and transitive.

Temporal setting:



- **Not symmetric!**
  There is a temporal path from A to B, but not from B to A.
- **Not transitive!**
  There is a temporal path from A to B and from B to C, but not from A to C.

Observation

Connectivity-related problems are computationally much harder in the temporal setting.

Minimum Temporal Spanners: Definition

Definition

A temporal graph $\mathscr{G}$ is (strictly or non-strictly) **temporally connected** (or **TC**) if there is a (strict or non-strict) temporal path between each pair of vertices (in both directions).

Minimum Temporal Spanners: Definition

Definition

A temporal graph $\mathscr{G}$ is (strictly or non-strictly) **temporally connected** (or **TC**) if there is a (strict or non-strict) temporal path between each pair of vertices (in both directions).

Minimum Temporal Spanner

Input: A TC temporal graph $\mathscr{G}$ and an integer k.
Task: Compute a TC spanning subgraph of $\mathscr{G}$ of size k (or decide that none exists).

Minimum Temporal Spanners: Definition

Definition

A temporal graph $\mathscr{G}$ is (strictly or non-strictly) **temporally connected** (or **TC**) if there is a (strict or non-strict) temporal path between each pair of vertices (in both directions).

Minimum Temporal Spanner

Input: A TC temporal graph $\mathscr{G}$ and an integer k.
Task: Compute a TC spanning subgraph of $\mathscr{G}$ of size k (or decide that none exists).

Size: number of time edges (min label) or number of underlying edges (min edge).

Minimum Temporal Spanners: Definition

Definition

A temporal graph $\mathscr{G}$ is (strictly or non-strictly) **temporally connected** (or **TC**) if there is a (strict or non-strict) temporal path between each pair of vertices (in both directions).

Minimum Temporal Spanner

Input: A TC temporal graph $\mathscr{G}$ and an integer k.
Task: Compute a TC spanning subgraph of $\mathscr{G}$ of size k (or decide that none exists).

Size: number of time edges (min label) or number of underlying edges (min edge).

4 Problem Variants: strict, non-strict $\times$ min label, min edge

Simple, Proper, and Happy Temporal Graphs

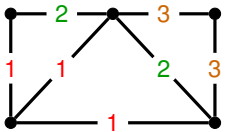
Simple Temporal Graphs

Only one label per edge!

Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

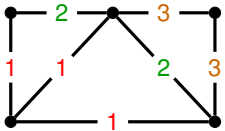
Only one label per edge!



Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

Only one label per edge!

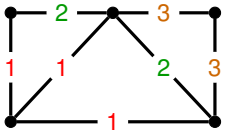


min label = min edge

Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

Only one label per edge!



min label = min edge

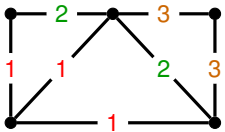
Proper Temporal Graphs

Different labels around each vertex!

Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

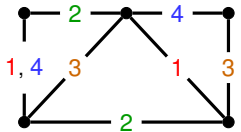
Only one label per edge!



min label = min edge

Proper Temporal Graphs

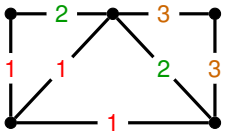
Different labels around each vertex!



Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

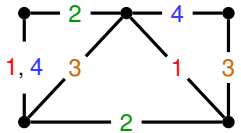
Only one label per edge!



min label = min edge

Proper Temporal Graphs

Different labels around each vertex!

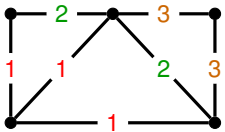


strict = non-strict

Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

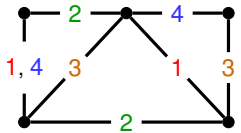
Only one label per edge!



min label = min edge

Proper Temporal Graphs

Different labels around each vertex!



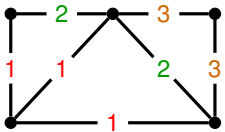
strict = non-strict

Happy Temporal Graphs: Simple and proper!

Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

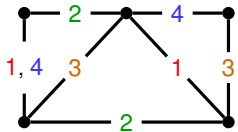
Only one label per edge!



min label = min edge

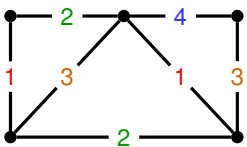
Proper Temporal Graphs

Different labels around each vertex!



strict = non-strict

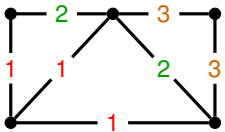
Happy Temporal Graphs: Simple and proper!



Simple, Proper, and Happy Temporal Graphs

Simple Temporal Graphs

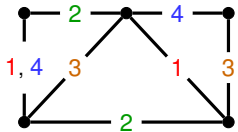
Only one label per edge!



min label = min edge

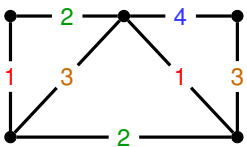
Proper Temporal Graphs

Different labels around each vertex!



strict = non-strict

Happy Temporal Graphs: Simple and proper!



All 4 problem variants are the same!

Known Results

NP-Hardness Results

Article	Setting	str / non-str	min label / edge
Axiotis and Fotakis [ICALP '16]	simple & not proper	non-strict	both
Akrida et al. [TOCS '17]	not simple & not proper	strict	min label
Casteigts et al. [SIROCCO '24]	not simple & proper	both	min edge
Our Work	simple & proper (happy)	**both**	**both**

Axiotis and Fotakis and Akrida et at. show also APX-hardness.

Known Results

NP-Hardness Results

Article	Setting	str / non-str	min label / edge
Axiotis and Fotakis [ICALP '16]	simple & not proper	non-strict	both
Akrida et al. [TOCS '17]	not simple & not proper	strict	min label
Casteigts et al. [SIROCCO '24]	not simple & proper	both	min edge
Our Work	simple & proper (happy)	**both**	**both**

Axiotis and Fotakis and Akrida et at. show also APX-hardness.

Algorithmic Results

Axiotis and Fotakis [ICALP '16] show poly-time solvability if underlying graph is a tree.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Work on related computational problems:

- Axiotis and Fotakis [ICALP '16] consider edge-weighted variant and give approximation results.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Work on related computational problems:

- Axiotis and Fotakis [ICALP '16] consider edge-weighted variant and give approximation results.
- Casteigts et al. [SIROCCO '24] consider variant where underlying graph of spanner is a tree.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Work on related computational problems:

- Axiotis and Fotakis [ICALP '16] consider edge-weighted variant and give approximation results.
- Casteigts et al. [SIROCCO '24] consider variant where underlying graph of spanner is a tree.
- Bilò et al. [ESA '22] consider spanners that approximately perserve distances.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Work on related computational problems:

- Axiotis and Fotakis [ICALP '16] consider edge-weighted variant and give approximation results.
- Casteigts et al. [SIROCCO '24] consider variant where underlying graph of spanner is a tree.
- Bilò et al. [ESA '22] consider spanners that approximately perserve distances.
- Bilò et al. [JCSS '24] consider blackout-tolerant spanners that approximately perserve distances.

Related Work

Long line of research on structural properties of temporal spanners (especially in so-called temporal cliques).

Axiotis and Fotakis [ICALP '16] showed that there are temporal spanners with $\Omega(n^2)$ edges.

Work on related computational problems:

- Axiotis and Fotakis [ICALP '16] consider edge-weighted variant and give approximation results.
- Casteigts et al. [SIROCCO '24] consider variant where underlying graph of spanner is a tree.
- Bilò et al. [ESA '22] consider spanners that approximately perserve distances.
- Bilò et al. [JCSS '24] consider blackout-tolerant spanners that approximately perserve distances.
- Kurita et al. [SAND '25] and Cioni et al. [IWOCA '26] study the enumeration of temporal spanners.

Our Results

Theorem

Minimum Temporal Spanner is NP-hard on happy temporal graphs.

Our Results

Theorem

Minimum Temporal Spanner is NP-hard on happy temporal graphs.

Corollary: Computing a 2-source spanner is NP-hard on happy temporal graphs.

Our Results

Theorem

Minimum Temporal Spanner is NP-hard on happy temporal graphs.

Corollary: Computing a 2-source spanner is NP-hard on happy temporal graphs.

Theorem

Minimum Temporal Spanner on happy temporal graphs is solvable in polynomial time if the underlying graph has a constant vertex cover number.

Our Results

Theorem

Minimum Temporal Spanner is NP-hard on happy temporal graphs.

Corollary: Computing a 2-source spanner is NP-hard on happy temporal graphs.

Theorem

Minimum Temporal Spanner on happy temporal graphs is solvable in polynomial time if the underlying graph has a constant vertex cover number.

Theorem

Minimum Temporal Spanner (min label) is W[1]-hard when parameterized by the feedback vertex number of the underlying graph on proper temporal graphs.

XP Algorithm for Vertex Cover Number I

Definition

A **temporal out-tree** T_v rooted at v is a minimal temporal graph in which there is a temporal path from v to each other vertex.

XP Algorithm for Vertex Cover Number I

Definition

A **temporal out-tree** T_v rooted at v is a minimal temporal graph in which there is a temporal path from v to each other vertex.

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

XP Algorithm for Vertex Cover Number I

Definition

A **temporal out-tree** T_v rooted at v is a minimal temporal graph in which there is a temporal path from v to each other vertex.

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

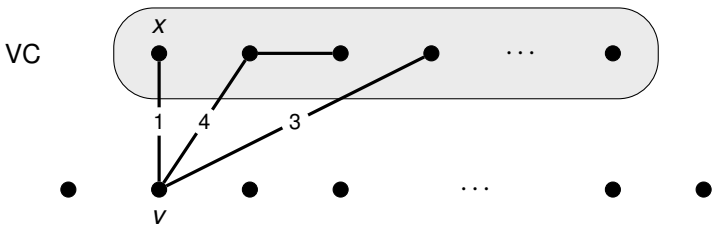
Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Proof of VC-Tree Lemma I

Let VC denote the vertex cover of the underlying graph. Consider a minimum spanner.

Proof of VC-Tree Lemma I

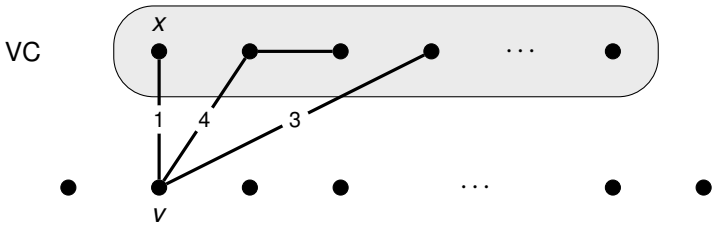
Let VC denote the vertex cover of the underlying graph. Consider a minimum spanner.



Let $v \in V_x$ for $x \in$ VC if v's time edge to x has the smallest label among time edges incident with v.

Proof of VC-Tree Lemma I

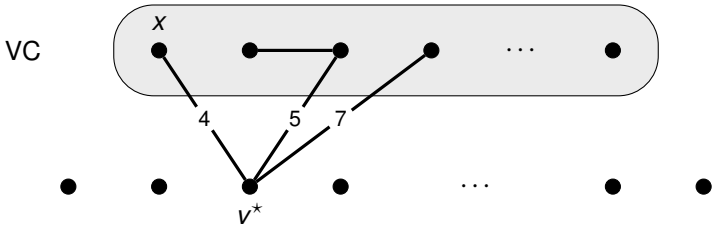
Let VC denote the vertex cover of the underlying graph. Consider a minimum spanner.



Let $v \in V_x$ for $x \in$ VC if v's time edge to x has the smallest label among time edges incident with v.

This forms a partition of the vertice in the independent set.

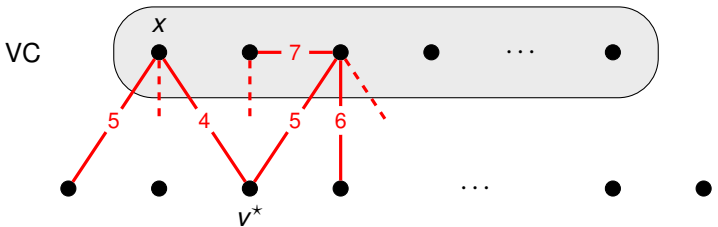
Proof of VC-Tree Lemma II



Let $v^\star \in V_x$ the vertex in V_x that has the largest label on its time edge to x among time edges between vertices in V_x and x.

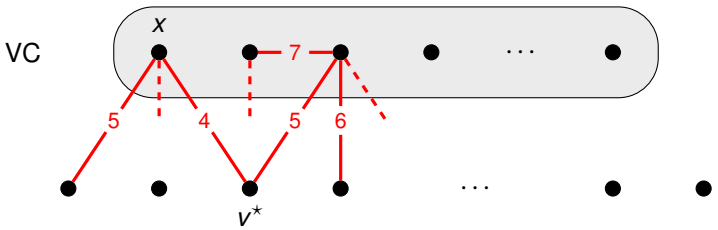
Proof of VC-Tree Lemma III

Let $T_{v^\star}$ be the temporal out-tree through which $v^\star$ reaches all other vertices.



Proof of VC-Tree Lemma III

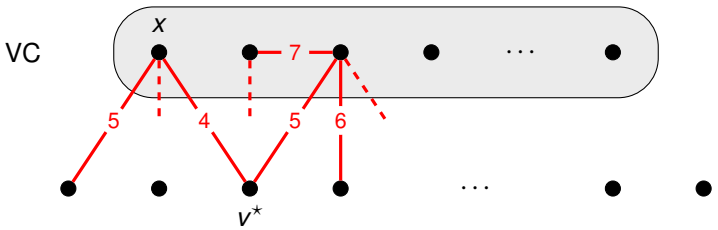
Let $T_{v^\star}$ be the temporal out-tree through which $v^\star$ reaches all other vertices.



We can re-root $T_{v^\star}$ to x to get a temporal out-tree for x. Let that temporal out-tree be T_x.

Proof of VC-Tree Lemma III

Let $T_{v^\star}$ be the temporal out-tree through which $v^\star$ reaches all other vertices.



We can re-root $T_{v^\star}$ to x to get a temporal out-tree for x. Let that temporal out-tree be T_x.

Each vertex in V_x can reach all other vertice by first reaching x via the direct edge, and then following T_x.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Algorithm:

- We compute a minimum vertex cover X for the underlying graph.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Algorithm:

- We compute a minimum vertex cover X for the underlying graph.
- We guess the temporal out-tree T_x for each $x \in X$.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Algorithm:

- We compute a minimum vertex cover X for the underlying graph.
- We guess the temporal out-tree T_x for each $x \in X$.
- We guess up to one extra edge incident with v for each $v \in V \setminus X$.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Algorithm:

- We compute a minimum vertex cover X for the underlying graph.
- We guess the temporal out-tree T_x for each $x \in X$.
- We guess up to one extra edge incident with v for each $v \in V \setminus X$.
- We define the spanner as the union of all guessed out-trees and additional edges.

XP Algorithm for Vertex Cover Number II

VC-Tree Lemma

Let $\mathscr{G}$ be a happy TC graph, let X be a vertex cover of the underlying graph with $|X| = d$. Let S be a minimum spanner of $\mathscr{G}$.

Then, S is a union of at most d temporal out-trees rooted in X and at most one extra edge incident to each vertex not in X.

Algorithm:

- We compute a minimum vertex cover X for the underlying graph.
- We guess the temporal out-tree T_x for each $x \in X$.
- We guess up to one extra edge incident with v for each $v \in V \setminus X$.
- We define the spanner as the union of all guessed out-trees and additional edges.
- We verify that the spanner is temporally connected and check whether its size is at most k.

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

Observation

The out-tree T_x for $x \in X$ has at most $2|X|$ inner vertices.

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

Observation

The out-tree T_x for $x \in X$ has at most $2|X|$ inner vertices.

- Guess inner vertices and how they connect.

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

Observation

The out-tree T_x for $x \in X$ has at most $2|X|$ inner vertices.

- Guess inner vertices and how they connect.
- For each remaining vertex (which is a leaf in T_x), guess to which inner vertex it connects.

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

Observation

The out-tree T_x for $x \in X$ has at most $2|X|$ inner vertices.

- Guess inner vertices and how they connect.
- For each remaining vertex (which is a leaf in T_x), guess to which inner vertex it connects.

Overall running time: $n^{O(|X|)}$, where n is the number of vertices.

XP Algorithm for Vertex Cover Number III

How to guess the temporal out-trees?

Observation

The out-tree T_x for $x \in X$ has at most $2|X|$ inner vertices.

- Guess inner vertices and how they connect.
- For each remaining vertex (which is a leaf in T_x), guess to which inner vertex it connects.

Overall running time: $n^{O(|X|)}$, where n is the number of vertices.

Correctness follows from the VC-Tree Lemma.

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Future Work:

- Can we get XP for vertex cover number on non-happy graphs?

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Future Work:

- Can we get XP for vertex cover number on non-happy graphs?
- Can we get XP with a smaller parameter?

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Future Work:

- Can we get XP for vertex cover number on non-happy graphs?
- Can we get XP with a smaller parameter?
- What can we do in directed temporal graphs?

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Future Work:

- Can we get XP for vertex cover number on non-happy graphs?
- Can we get XP with a smaller parameter?
- What can we do in directed temporal graphs?

Link to arXiv.

Conclusion and Future Research

Summary:

- **Minimum Temporal Spanner** is NP-hard on happy temporal graphs.
- **Minimum Temporal Spanner** in XP when param. by the vertex cover number of the underlying graph on happy temporal graphs.
- Conjecture (ongoing work): **Minimum Temporal Spanner** is W[1]-hard when param. by the vertex cover number of the underlying graph on happy temporal graphs.

Future Work:

- Can we get XP for vertex cover number on non-happy graphs?
- Can we get XP with a smaller parameter?
- What can we do in directed temporal graphs?

Link to arXiv.

Thank you!

Happy NP-Hardness