

# Some Algorithmic Problems on Temporal Graphs

Paul G. Spirakis

Department of Computer Science,  
University of Liverpool UK

9th International Conference on Algebraic Informatics  
CAI, 29 Oct 2022

# Static and Temporal Graphs

Modern networks are **highly dynamic**:

- **Social networks**: friendships are added/removed, individuals **leave**, new ones **enter**
- **Transportation networks**: transportation units change with time their **position** in the network
- **Physical systems**: e.g. systems of **interacting particles**

The common characteristic in all these applications:

- the **graph topology** is subject to **discrete changes** over time
- ⇒ the notion of **vertex adjacency** must be appropriately re-defined (by introducing the **time dimension** in the graph definition)

Various graph concepts (e.g. reachability, connectivity):

- crucially **depend** on the **exact temporal ordering** of the **edges**

# Temporal graphs (formally)

## Definition (Temporal Graph)

A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- 
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

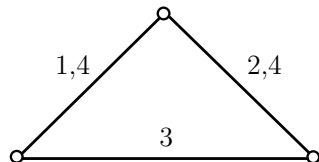
# Temporal graphs (formally)

## Definition (Temporal Graph)

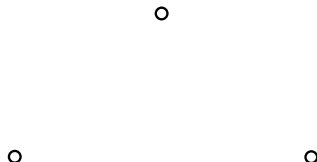
A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- 
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

temporal graph:



temporal instances:



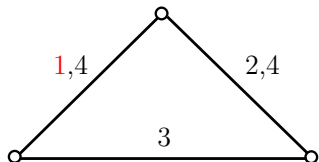
# Temporal graphs (formally)

## Definition (Temporal Graph)

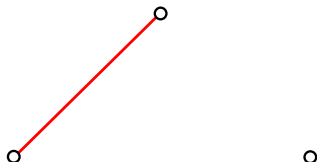
A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

temporal graph:



temporal instances:



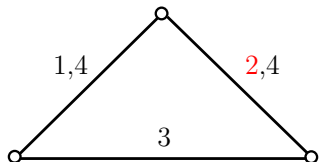
# Temporal graphs (formally)

## Definition (Temporal Graph)

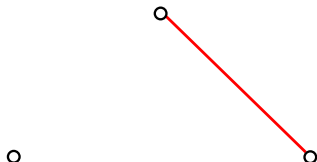
A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- 
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

temporal graph:



temporal instances:



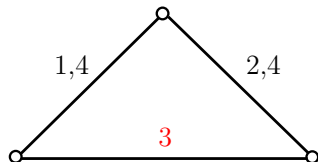
# Temporal graphs (formally)

## Definition (Temporal Graph)

A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- 
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

temporal graph:



temporal instances:



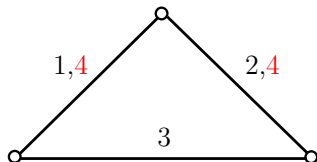
# Temporal graphs (formally)

## Definition (Temporal Graph)

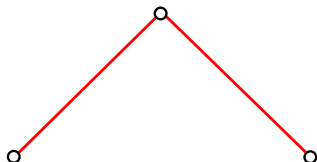
A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- If  $t \in \lambda(e)$  then edge  $e$  is **available** at time  $t$

temporal graph:



temporal instances:





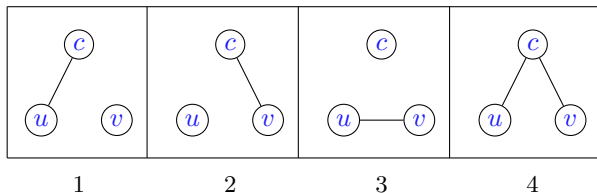
# Temporal graphs (formally)

## Definition (Temporal Graph)

A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
- $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.

Alternatively, we can view it as a **sequence** of static graphs, the **snapshots**:



# Temporal graphs (formally)

## Definition (Temporal Graph)

A **temporal graph** is a pair  $(G, \lambda)$  where:

- $G = (V, E)$  is an **underlying (di)graph** and
  - $\lambda : E \rightarrow 2^{\mathbb{N}}$  is a discrete **time-labeling** function.
- 
- Usually the input is a graph  $G$  with given labels  $\{\lambda(e) : e \in E\}$
  - Other models have been studied as well, e.g. various randomized models for temporal graphs:
    - every edge of  $G$  gets  $\rho$  random labels in each period  $\alpha$  of time [Akrida, Gasieniec, Mertzios, Spirakis, *J. Par. Distr. Comp.*, 2016]
    - every edge appears according to a probability distribution [Akrida, Mertzios, Nikolettseas, Raptopoulos, Spirakis, Zamaraev, *J. Computer and System Sciences*, 2020]
    - random (temporal) edge permutation in an Erdős-Renyi random graph [Casteigts, Raskin, Renken, Zamaraev, *FOCS*, 2021]

- Temporal graphs
- Temporal parameters and temporal paths: a warm-up
- Temporal vertex cover
- Temporal transitive orientations
- Stochastic temporal graphs

# Temporal paths

The conceptual shift from static to temporal graphs significantly impacts:

- the **definition** of basic **graph parameters**
- the **type** of **tasks** to be computed

Graph properties can be classified as:

- **a-temporal**, i.e. satisfied **at every instance**
  - connectivity at every point in time
- **temporal**, i.e. satisfied **over time**
  - communication routes over time

# Temporal paths

The most natural known **temporal notion** in temporal graphs:

**Definition (Temporal path; Time-respecting path; Journey)**

Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ . A **temporal path** is a sequence  $((e_1, \ell_1), (e_2, \ell_2), \dots, (e_k, \ell_k))$ , where:

$$\ell_1 < \ell_2 < \dots < \ell_k$$

and  $\ell_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ . A **temporal path** is a sequence  $((e_1, \ell_1), (e_2, \ell_2), \dots, (e_k, \ell_k))$ , where:

$$\ell_1 < \ell_2 < \dots < \ell_k$$

and  $\ell_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

Motivation due to **causality** in information dissemination:

- information “**flows**” along edges whose labels respect **time ordering**
- ⇒ strictly increasing labels along the path
- a “static path” given “in pieces”

# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

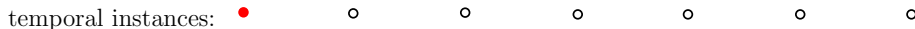
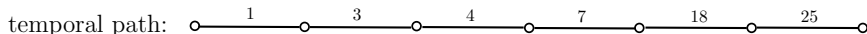
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ . A **temporal path** is a sequence  $((e_1, \ell_1), (e_2, \ell_2), \dots, (e_k, \ell_k))$ , where:

$$\ell_1 < \ell_2 < \dots < \ell_k$$

and  $\ell_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

Motivation due to **causality** in information dissemination:

- information “flows” along edges whose labels respect **time ordering**
- ⇒ strictly increasing labels along the path
- a “static path” given “in pieces”
- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

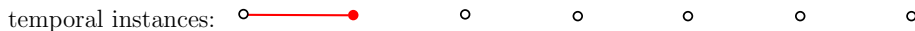
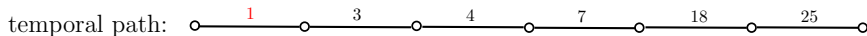
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:





# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

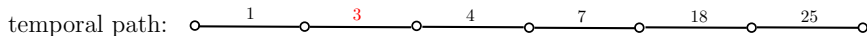
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

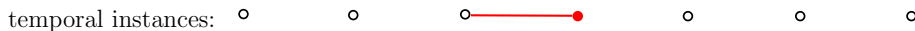
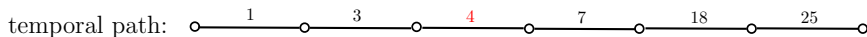
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

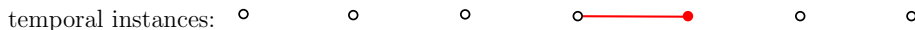
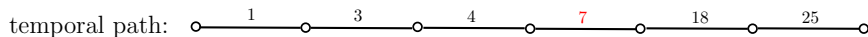
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

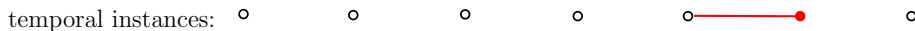
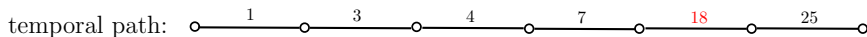
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

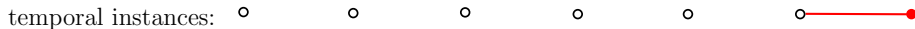
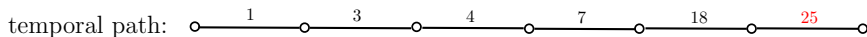
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, \ell_1), (e_2, \ell_2), \dots, (e_k, \ell_k))$ , where:

$$\ell_1 < \ell_2 < \dots < \ell_k$$

and  $\ell_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

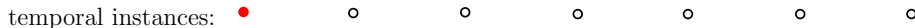
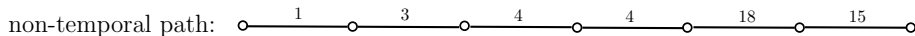
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

**Definition** (Temporal path; Time-respecting path; Journey)

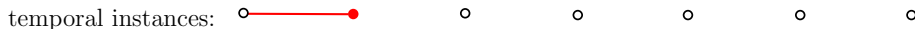
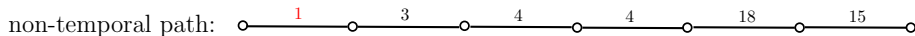
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

**Definition** (Temporal path; Time-respecting path; Journey)

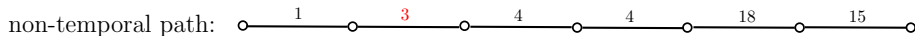
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:





# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

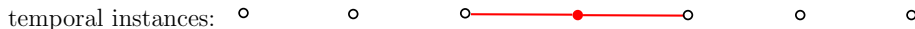
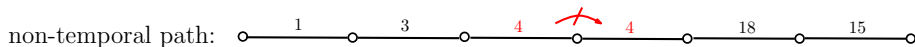
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

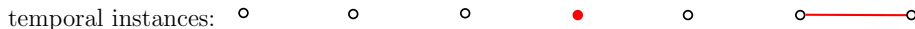
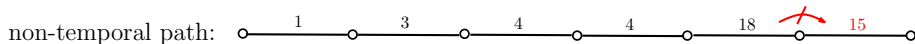
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

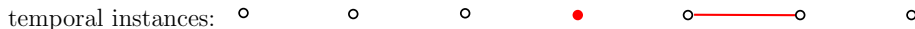
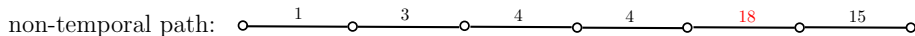
Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ .

A **temporal path** is a sequence  $((e_1, l_1), (e_2, l_2), \dots, (e_k, l_k))$ , where:

$$l_1 < l_2 < \dots < l_k$$

and  $l_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

- A **non-temporal path**:



# Temporal paths

The most natural known **temporal notion** in temporal graphs:

## Definition (Temporal path; Time-respecting path; Journey)

Let  $(G, \lambda)$  be a **temporal graph** and  $P = (e_1, e_2, \dots, e_k)$  be a walk in  $G$ . A **temporal path** is a sequence  $((e_1, \ell_1), (e_2, \ell_2), \dots, (e_k, \ell_k))$ , where:

$$\ell_1 < \ell_2 < \dots < \ell_k$$

and  $\ell_i \in \lambda(e_i)$ ,  $1 \leq i \leq k$ .

A temporal path can also be considered to be **non-strict** if we demand that:

$$\ell_1 \leq \ell_2 \leq \dots \leq \ell_k$$

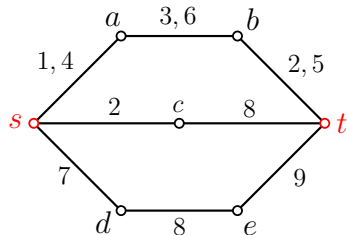
# Metrics to optimize

**Question:** What is the **temporal analogue** of an  **$s$ - $t$  shortest path**?

**Answer:** Not uniquely defined!

- **topologically shortest** path: smallest **number of edges**
- **fastest** path: smallest **duration**
- **foremost** path: smallest **arrival time**

Example:



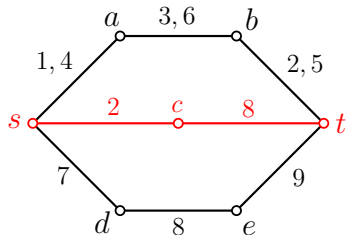
# Metrics to optimize

**Question:** What is the **temporal analogue** of an ***s-t* shortest path**?

**Answer:** Not uniquely defined!

- **topologically shortest** path: smallest **number of edges**
- **fastest** path: smallest **duration**
- **foremost** path: smallest **arrival time**

Example:



shortest:  $s-c-t$  (two edges)

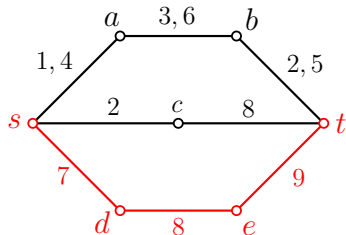
# Metrics to optimize

**Question:** What is the **temporal analogue** of an  **$s$ - $t$  shortest path**?

**Answer:** Not uniquely defined!

- **topologically shortest** path: smallest **number of edges**
- **fastest** path: smallest **duration**
- **foremost** path: smallest **arrival time**

Example:



shortest:  $s-c-t$  (two edges)

fastest:  $s-d-e-t$  (no intermediate waiting)

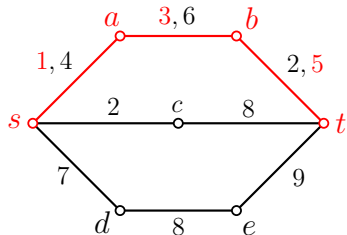
# Metrics to optimize

**Question:** What is the **temporal analogue** of an ***s-t* shortest path**?

**Answer:** Not uniquely defined!

- **topologically shortest** path: smallest **number of edges**
- **fastest** path: smallest **duration**
- **foremost** path: smallest **arrival time**

Example:



shortest:  $s-c-t$  (two edges)

fastest:  $s-d-e-t$  (no intermediate waiting)

foremost:  $s-a-b-t$  (arriving at time 6)



- Temporal graphs
- Temporal paths: a warm-up
- Temporal vertex cover
- Temporal transitive orientations
- Stochastic temporal graphs

# Basic definitions I

To specify a **temporal graph class**, we can:

- either restrict the **underlying graph**  $G$ ,
- or restrict the **labeling**  $\lambda : E \rightarrow 2^{\mathbb{N}}$  (or both)

# Basic definitions I

To specify a **temporal graph class**, we can:

- either restrict the **underlying graph**  $G$ ,
- or restrict the **labeling**  $\lambda : E \rightarrow 2^{\mathbb{N}}$  (or both)

## Definition (Temporal Graph Classes)

For a class  $\mathcal{X}$  of static graphs we say that a temporal graph  $(G, \lambda)$  is

- **$\mathcal{X}$  temporal**, if  $G \in \mathcal{X}$ ;
- **always  $\mathcal{X}$  temporal**, if  $G_i \in \mathcal{X}$  for every  $i \in [T] = \{1, 2, \dots, T\}$ .

# Basic definitions I

To specify a **temporal graph class**, we can:

- either restrict the **underlying graph**  $G$ ,
- or restrict the **labeling**  $\lambda : E \rightarrow 2^{\mathbb{N}}$  (or both)

## Definition (Temporal Graph Classes)

For a class  $\mathcal{X}$  of static graphs we say that a temporal graph  $(G, \lambda)$  is

- **$\mathcal{X}$  temporal**, if  $G \in \mathcal{X}$ ;
- **always  $\mathcal{X}$  temporal**, if  $G_i \in \mathcal{X}$  for every  $i \in [T] = \{1, 2, \dots, T\}$ .

## Definition (Temporal Vertex Subset)

A pair  $(u, t) \in V \times [T]$  is called the **appearance of vertex  $u$  at time  $t$** .

A **temporal vertex subset** of  $(G, \lambda)$  is a set  $\mathcal{S} \subseteq V \times [T]$  of vertex appearances in  $(G, \lambda)$ .

## Definition (Edge is Temporally Covered)

A vertex appearance  $(w, t)$  **temporally covers** an edge  $e$  if:

- (i)  $w$  covers  $e$ , i.e.  $w \in e$ , and
- (ii)  $t \in \lambda(e)$ , i.e. the edge  $e$  is **active** during the time slot  $t$ .

[Akrida, Mertzios, Spirakis, Zamaraev, *J. Comp. & System Sciences*, 2020]

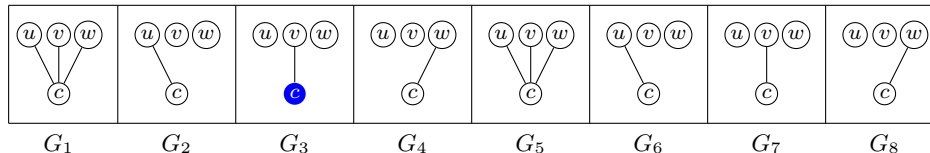
# Basic definitions II

## Definition (Edge is Temporally Covered)

A vertex appearance  $(w, t)$  **temporally covers** an edge  $e$  if:

- (i)  $w$  covers  $e$ , i.e.  $w \in e$ , and
- (ii)  $t \in \lambda(e)$ , i.e. the edge  $e$  is **active** during the time slot  $t$ .

### Example:



- $(c, 3)$  **temporally covers** edge  $cv$ , but
- $(c, 3)$  **temporally covers** neither  $cu$ , nor  $cw$ .

[Akrida, Mertzios, Spirakis, Zamaraev, *J. Comp. & System Sciences*, 2020]

## Definition (Temporal Vertex Cover)

A **temporal vertex cover** of  $(G, \lambda)$  is a temporal vertex subset  $\mathcal{S}$  of  $(G, \lambda)$  such that every edge  $e \in E(G)$  is **temporally covered** by at least one vertex appearance in  $\mathcal{S}$ .

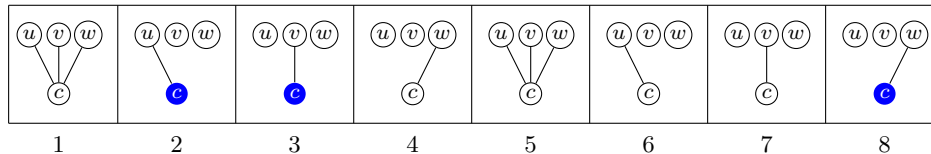
[Akrida, Mertzios, Spirakis, Zamaraev, *J. Comp. & System Sciences*, 2020]

# Basic definitions: Temporal Vertex Cover

## Definition (Temporal Vertex Cover)

A **temporal vertex cover** of  $(G, \lambda)$  is a temporal vertex subset  $\mathcal{S}$  of  $(G, \lambda)$  such that every edge  $e \in E(G)$  is **temporally covered** by at least one vertex appearance in  $\mathcal{S}$ .

## Example



–  $\{(c, 2), (c, 3), (c, 8)\}$  is a Temporal Vertex Cover

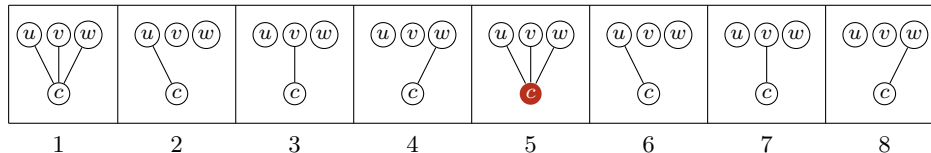


# Basic definitions: Temporal Vertex Cover

## Definition (Temporal Vertex Cover)

A **temporal vertex cover** of  $(G, \lambda)$  is a temporal vertex subset  $\mathcal{S}$  of  $(G, \lambda)$  such that every edge  $e \in E(G)$  is **temporally covered** by at least one vertex appearance in  $\mathcal{S}$ .

## Example



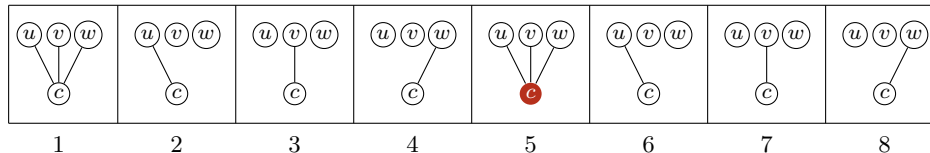
- $\{(c, 2), (c, 3), (c, 8)\}$  is a Temporal Vertex Cover
- $\{(c, 5)\}$  is a **minimum** Temporal Vertex Cover

# Basic definitions: Temporal Vertex Cover

## Definition (Temporal Vertex Cover)

A **temporal vertex cover** of  $(G, \lambda)$  is a temporal vertex subset  $\mathcal{S}$  of  $(G, \lambda)$  such that every edge  $e \in E(G)$  is **temporally covered** by at least one vertex appearance in  $\mathcal{S}$ .

## Example



## TEMPORAL VERTEX COVER (TVC)

**Input:** A temporal graph  $(G, \lambda)$ .

**Output:** A **temporal vertex cover**  $\mathcal{S}$  of  $(G, \lambda)$  with the minimum  $|\mathcal{S}|$ .

## Definition (Time Windows)

- 1 For every time slot  $t \in [1, T - \Delta + 1]$ :  
the **time window**  $W_t = [t, t + \Delta - 1]$  is the sequence of the  $\Delta$  consecutive time slots  $t, t + 1, \dots, t + \Delta - 1$ .

## Definition (Time Windows)

- 1 For every time slot  $t \in [1, T - \Delta + 1]$ :  
the **time window**  $W_t = [t, t + \Delta - 1]$  is the sequence of the  $\Delta$  consecutive time slots  $t, t + 1, \dots, t + \Delta - 1$ .
- 2  $E[W_t] = \bigcup_{i \in W_t} E_i$  is the union of all edges appearing at least once in the time window  $W_t$ .
- 3  $\mathcal{S}[W_t] = \{(w, t) \in \mathcal{S} : t \in W_t\}$  is the restriction of the temporal vertex subset  $\mathcal{S}$  to the window  $W_t$ .

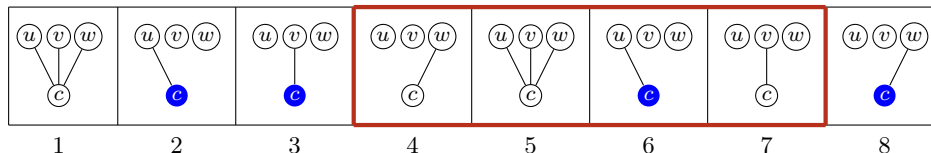
## Definition (Sliding $\Delta$ -Window Temporal Vertex Cover)

A **sliding  $\Delta$ -window temporal vertex cover** of  $(G, \lambda)$  is a temporal vertex subset  $\mathcal{S}$  of  $(G, \lambda)$  such that:

- for every time window  $W_t$  and for every edge  $e \in E[W_t]$ ,
- $e$  is **temporally covered** by at least one vertex appearance  $(w, t) \in \mathcal{S}[W_t]$ .

# Basic definitions: Sliding Window Temporal Vertex Cover

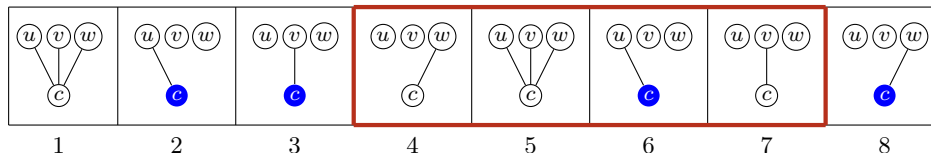
## Example ( $\Delta = 4$ )



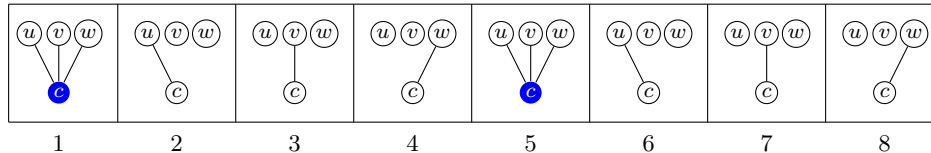
- $\{(c, 2), (c, 3), (c, 6), (c, 8)\}$  is **not** a sliding  $\Delta$ -window temporal vertex cover, as edges  $cv, cw \in E[W_4]$  are **not** temporally covered in **window**  $W_4$ .

# Basic definitions: Sliding Window Temporal Vertex Cover

## Example ( $\Delta = 4$ )



- $\{(c, 2), (c, 3), (c, 6), (c, 8)\}$  is **not** a sliding  $\Delta$ -window temporal vertex cover, as edges  $cv, cw \in E[W_4]$  are **not** temporally covered in window  $W_4$ .



- $\{(c, 1), (c, 5)\}$  is a **sliding  $\Delta$ -window temporal vertex cover**.

## SLIDING WINDOW TEMPORAL VERTEX COVER (SW-TVC)

**Input:** A temporal graph  $(G, \lambda)$  with lifetime  $T$ , and an integer  $\Delta \leq T$ .

**Output:** A sliding  $\Delta$ -window temporal vertex cover  $\mathcal{S}$  of  $(G, \lambda)$  with the minimum  $|\mathcal{S}|$ .

Motivation:

- **(static) Vertex Cover:**  
network surveillance (e.g. CCTV cameras etc.)
- **Temporal Vertex Cover:**  
network surveillance in a dynamic network
- **Sliding Window Temporal Vertex Cover:**  
dynamic surveillance in every possible  $\Delta$ -time window  
(e.g. for crimes that need time  $\Delta$  to be performed)



# Temporal Vertex Cover: the star temporal case

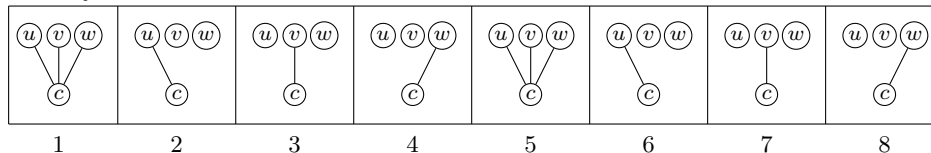
Lemma (Akrida et al., *J. Comp. & System Sciences*, 2020)

**TVC** on *star temporal graphs* is equivalent to **SET COVER**.

- **leafs** of the underlying star  $\leftrightarrow$  **ground set** of the **SET COVER** instance
- each **snapshot** graph  $\leftrightarrow$  a **set** in the **SET COVER** instance

**Goal:** Choose **sets** (**snapshots**) to cover all **elements** (**leafs' edges**)

**Example:**



# Temporal Vertex Cover: the star temporal case

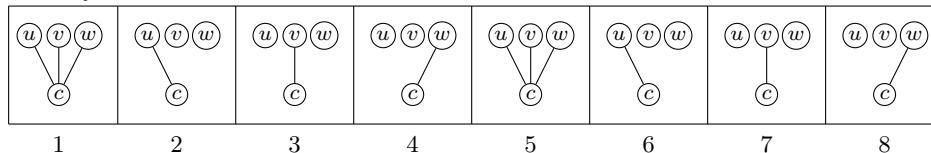
Lemma (Akrida et al., *J. Comp. & System Sciences*, 2020)

**TVC** on *star temporal graphs* is equivalent to **SET COVER**.

- **leafs** of the underlying star  $\leftrightarrow$  **ground set** of the **SET COVER** instance
- each **snapshot** graph  $\leftrightarrow$  a **set** in the **SET COVER** instance

Goal: Choose **sets** (snapshots) to cover all **elements** (leafs' edges)

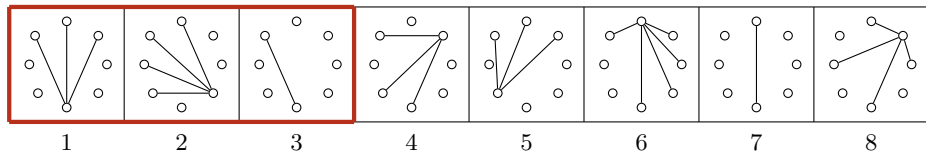
## Example:



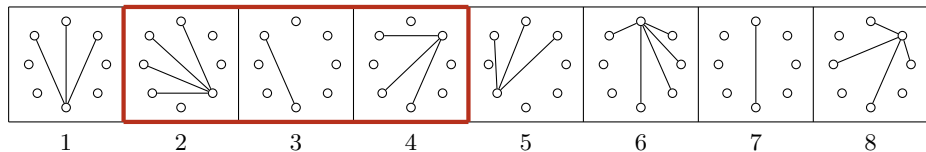
1 **Universe:**  $\{u, v, w\}$

2 **Sets:**  $S_1 = \{u, v, w\}$ ,  $S_2 = \{u\}$ ,  $S_3 = \{v\}$ ,  $S_4 = \{w\}$ , ...

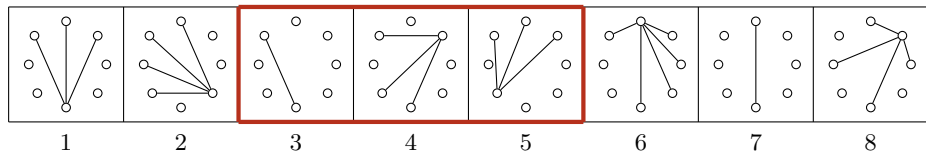
# SW-TVC: **always star** temporal graphs



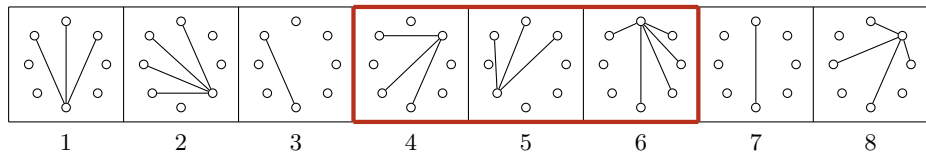
# SW-TVC: always star temporal graphs



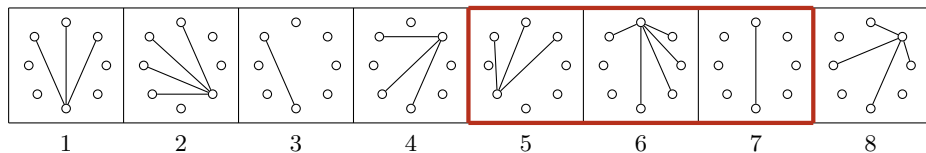
# SW-TVC: **always star** temporal graphs



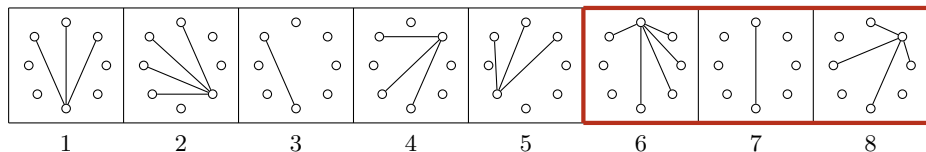
# SW-TVC: **always star** temporal graphs



# SW-TVC: **always star** temporal graphs



## SW-TVC: always star temporal graphs

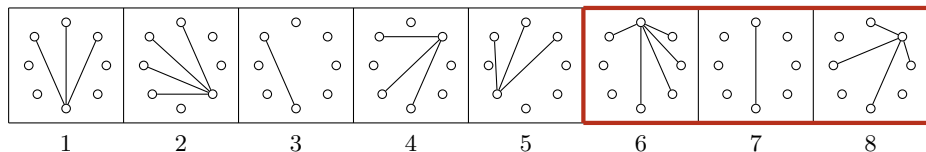


- On **always star** temporal graphs, a minimum size **SW-TVC** contains **at most one vertex** (the **star center**) in each snapshot

⇒ we assign a Boolean variable  $x_i \in \{0, 1\}$  for the snapshot at time  $i$



## SW-TVC: always star temporal graphs



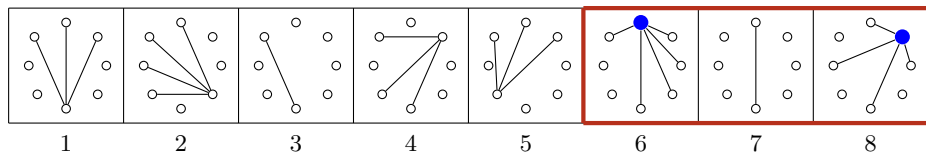
- On **always star** temporal graphs, a minimum size **SW-TVC** contains **at most one vertex** (the **star center**) in each snapshot

⇒ we assign a Boolean variable  $x_i \in \{0, 1\}$  for the snapshot at time  $i$

- For variables  $x_1, x_2, \dots, x_\Delta$  we define  $f(t; x_1, x_2, \dots, x_\Delta)$  to be the smallest cardinality of a sliding  **$\Delta$ -window temporal vertex cover  $\mathcal{S}$**  of  $(G, \lambda)|_{[1, t+\Delta-1]}$ , such that the solution in the time window  $W_t = \{t, \dots, t + \Delta - 1\}$  is given by the **variables  $x_1, x_2, \dots, x_\Delta$** .

# SW-TVC: always star temporal graphs

$$f(6; 1, 0, 1)$$

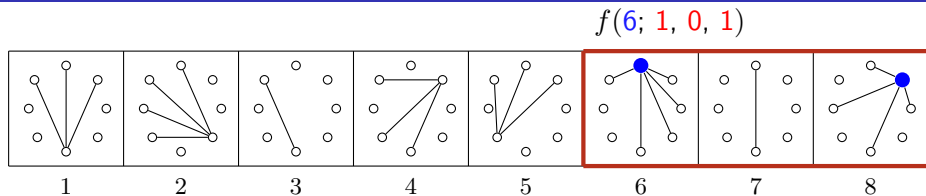


- On **always star** temporal graphs, a minimum size **SW-TVC** contains **at most one vertex** (the **star center**) in each snapshot

⇒ we assign a Boolean variable  $x_i \in \{0, 1\}$  for the snapshot at time  $i$

- For variables  $x_1, x_2, \dots, x_\Delta$  we define  $f(t; x_1, x_2, \dots, x_\Delta)$  to be the smallest cardinality of a sliding  **$\Delta$ -window temporal vertex cover**  $\mathcal{S}$  of  $(G, \lambda)|_{[1, t+\Delta-1]}$ , such that the solution in the time window  $W_t = \{t, \dots, t + \Delta - 1\}$  is given by the **variables**  $x_1, x_2, \dots, x_\Delta$ .

# SW-TVC: always star temporal graphs



- On **always star** temporal graphs, a minimum size **SW-TVC** contains **at most one vertex** (the **star center**) in each snapshot

⇒ we assign a Boolean variable  $x_i \in \{0, 1\}$  for the snapshot at time  $i$

- For variables  $x_1, x_2, \dots, x_\Delta$  we define  $f(t; x_1, x_2, \dots, x_\Delta)$  to be the smallest cardinality of a sliding  **$\Delta$ -window temporal vertex cover**  $\mathcal{S}$  of  $(G, \lambda)|_{[1, t+\Delta-1]}$ , such that the solution in the time window  $W_t = \{t, \dots, t + \Delta - 1\}$  is given by the **variables**  $x_1, x_2, \dots, x_\Delta$ .

Lemma (dynamic programming)

$$f(t; x_1, x_2, \dots, x_\Delta) = x_\Delta + \min_{y \in \{0,1\}} \{f(t-1; y, x_1, x_2, \dots, x_{\Delta-1})\}$$

Theorem (always star temporal graphs)

**SW-TVC** on *always star temporal graphs* can be solved in  $O(T\Delta(n + m) \cdot 2^\Delta)$  time.

## Theorem (always star temporal graphs)

**SW-TVC** on *always star temporal graphs* can be solved in  $O(T\Delta(n+m) \cdot 2^\Delta)$  time.

## Theorem (the general case)

**SW-TVC** on *general temporal graphs* can be solved in  $O(T\Delta(n+m) \cdot 2^{n(\Delta+1)})$  time.

Main idea:

- for each of the  $\Delta$  snapshots in the (currently) last  $\Delta$ -window, we enumerate all  $2^n$  vertex subsets,
- instead of just enumerating over the truth values of  $\Delta$  Boolean variables (“always star” case)

## Theorem (always star temporal graphs)

**SW-TVC** on *always star temporal graphs* can be solved in  $O(T\Delta(n+m) \cdot 2^\Delta)$  time.

## Theorem (the general case)

**SW-TVC** on *general temporal graphs* can be solved in  $O(T\Delta(n+m) \cdot 2^{n(\Delta+1)})$  time.

We can prove:

## Corollary

Our  $O(T\Delta(n+m) \cdot 2^{n(\Delta+1)})$ -time algorithm is asymptotically almost optimal (assuming ETH).

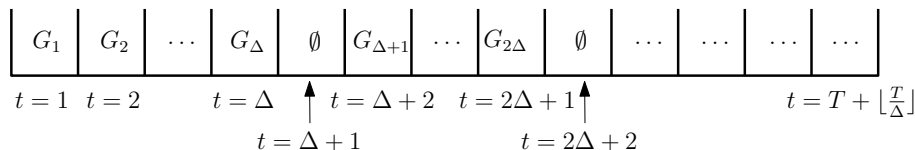
If the parameter  $\Delta$  (the size of a sliding window) is **fixed**, we refer to **SW-TVC** as  **$\Delta$ -TVC** (i.e.  $\Delta$  is a part of the problem name).

# $\Delta$ -TVC

If the parameter  $\Delta$  (the size of a sliding window) is **fixed**, we refer to **SW-TVC** as  **$\Delta$ -TVC** (i.e.  $\Delta$  is a part of the problem name).

## Observation

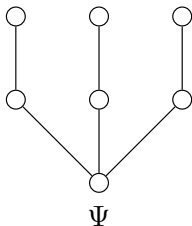
**$(\Delta + 1)$ -TVC** is at least as hard as  **$\Delta$ -TVC**.





## 2-TVC for $\max \text{deg} \leq 3$

Let  $\mathcal{X}$  be the class of graphs whose connected components are induced subgraphs of graph  $\Psi$ , with **maximum degree 3**:



Clearly, **VERTEX COVER** is **linearly solvable** on graphs from  $\mathcal{X}$ .

Theorem (Akrida et al., *J. Comp. & System Sciences*, 2020)

There is **no PTAS** for **2-TVC** on **always  $\mathcal{X}$**  temporal graphs.

- What is the complexity for (always) **maximum degree 2** ?

## 2-TVC for $\max \text{deg} \leq 2$

Our results when the underlying graph is a path or a cycle:

- **linear-time** algorithm for **TVC** (no sliding windows)
- **2-TVC** is **NP-hard**
- **PTAS** for  **$\Delta$ -TVC**, for any  $\Delta \geq 2$

[Hamm, Klobas, Mertzios, Spirakis, *AAAI*, 2022]

## 2-TVC for $\max \text{deg} \leq 2$

Greedy linear-time algorithm for TVC on paths:

- visit the vertices from left to right

**for** every  $i = 1, 2, \dots, n - 1$  **do**

**if**  $e_i$  and  $e_{i+1}$  appear\* at the same time  $t$  (for some  $t$ ) **then**

add  $(v_{i+1}, t)$  to  $\mathcal{C}$  (where  $e_i \cap e_{i+1} = \{v_{i+1}\}$ )

$i = i + 2$

**else**

Add to  $\mathcal{C}$  an arbitrary  $(v_i, t)$  or  $(v_{i+1}, t)$ , where  $t \in \lambda(e_i)$ .

$i = i + 1$

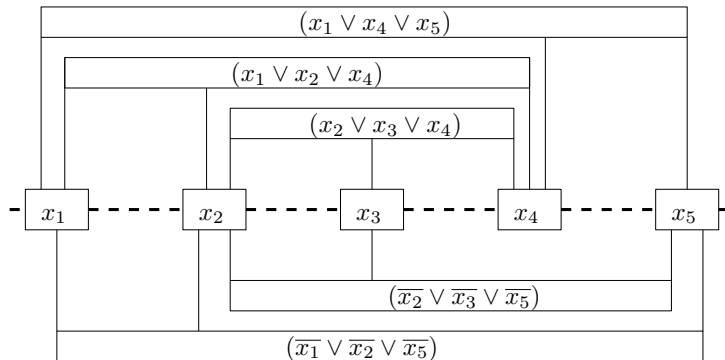
**return**  $\mathcal{C}$ .

\* Denote by  $e_i = v_i v_{i+1}$ , for every  $i = 1, 2, \dots, n - 1$ .

## 2-TVC is NP-hard on temporal paths

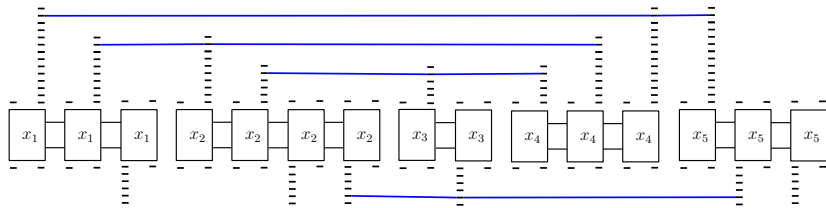
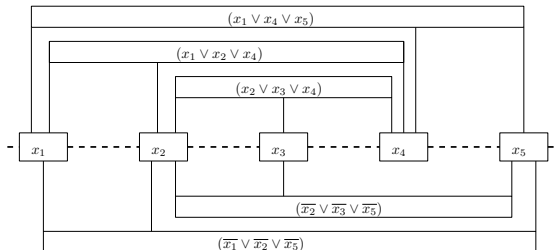
Reduction from **planar monotone rectilinear 3SAT**.

$$\phi = (x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_4) \wedge (x_1 \vee x_4 \vee x_5) \wedge \\ (\overline{x_2} \vee \overline{x_3} \vee \overline{x_5}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_5})$$



# 2-TVC is NP-hard on temporal paths

High-level construction:



Reduction to this problem:

## GEOMETRIC HITTING SET

**Input:** A pair  $R = (P, D)$  (range space), where  $P$  is a set of points in  $\mathbb{R}^2$  and  $D$  is a set of regions covering all points of  $P$ .

**Output:** A smallest subset of points  $S \subseteq P$ , such that every region in  $D$  contains at least one point of  $S$ .

# PTAS for $\Delta$ -TVC on temporal paths

Reduction to this problem:

## GEOMETRIC HITTING SET

**Input:** A pair  $R = (P, D)$  (range space), where  $P$  is a set of points in  $\mathbb{R}^2$  and  $D$  is a set of regions covering all points of  $P$ .

**Output:** A smallest subset of points  $S \subseteq P$ , such that every region in  $D$  contains at least one point of  $S$ .

PTAS for  $r$ -admissible set regions:

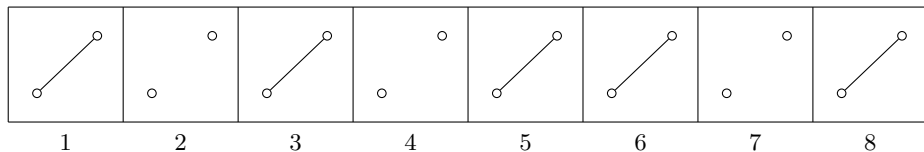
- boundaries of  $s_1, s_2 \in D$  intersect at most  $r$  times
- $s_1 \setminus s_2$  and  $s_2 \setminus s_1$  are connected regions

Theorem (Mustafa and Ray, *Discrete and Computat. Geometry*, 2010)

For every  $\varepsilon > 0$ , there is an  $(1 + \varepsilon)$ -approximation algorithm for GEOMETRIC HITTING SET that runs in  $O(|D||P|^{O(\varepsilon^{-2})})$  time.

# SW-TVC: approximation algorithms II

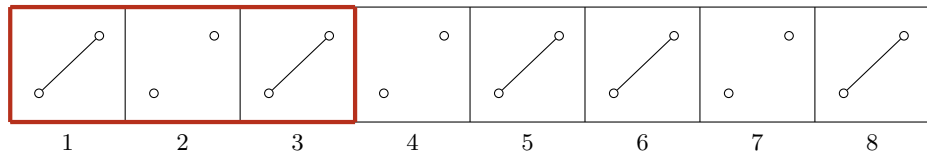
Single-edge temporal graph: **exact algorithm**



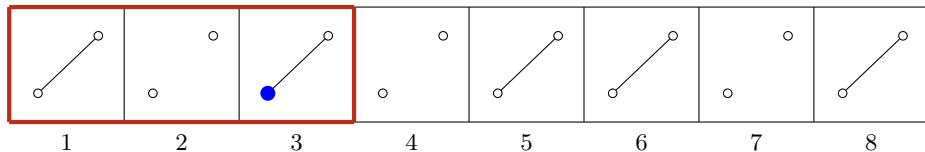


# SW-TVC: approximation algorithms II

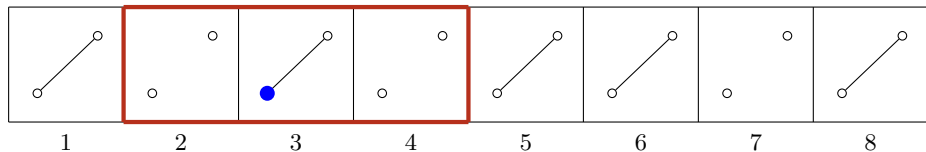
Single-edge temporal graph: **exact algorithm**



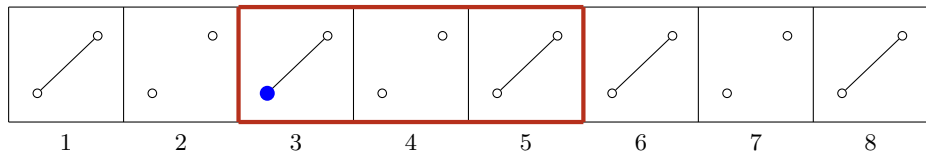
## Single-edge temporal graph: **exact algorithm**



## Single-edge temporal graph: **exact algorithm**

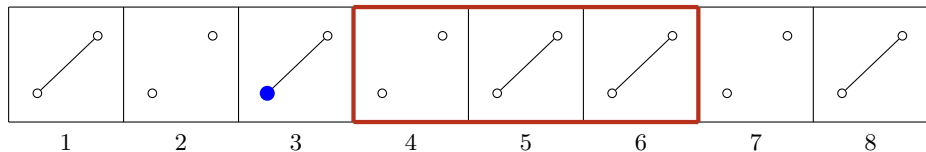


## Single-edge temporal graph: **exact algorithm**

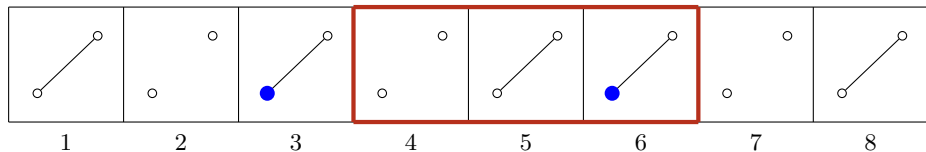


# SW-TVC: approximation algorithms II

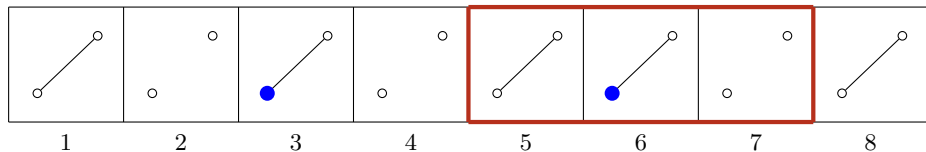
Single-edge temporal graph: **exact algorithm**



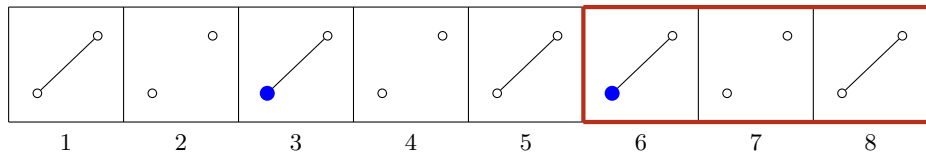
## Single-edge temporal graph: **exact algorithm**



## Single-edge temporal graph: **exact algorithm**



## Single-edge temporal graph: **exact algorithm**





## Single-edge temporal graph: **exact algorithm**

- 1 In the first window  $W_t = [1, \Delta]$ : cover the edge at the latest time slot it appears  
(to “cover” as many other windows as possible)
  - 2 Remove all windows that are now covered
  - 3 Repeat
- greedy algorithm
  - linear time

# SW-TVC: approximation algorithms II

**Always degree at most  $d$  temp. graphs:  $d$ -approx. algorithm**

Main idea:

- solve independently each single-edge subgraph of  $G$
- take the union of the solutions

# SW-TVC: approximation algorithms II

Always degree at most  $d$  temp. graphs:  **$d$ -approx. algorithm**

Main idea:

- solve independently each single-edge subgraph of  $G$
- take the union of the solutions

Lemma (Akrida et al., *J. Comp. & System Sciences*, 2020)

There is a  $O(mT)$ -time  $d$ -approximation algorithm for SW-TVC on always degree at most  $d$  temporal graphs.

- Can we do better?

# SW-TVC: approximation algorithms II

**Always degree at most  $d$  temp. graphs:**  
 **$(d - 1)$ -approx. algorithm**

Main idea:

- instead of single edges, solve first SW-TVC independently **every possible  $P_3$**  in  $(G, \lambda)$
- take the **union** of the solutions

# SW-TVC: approximation algorithms II

**Always degree at most  $d$  temp. graphs:**  
 **$(d - 1)$ -approx. algorithm**

Main idea:

- instead of single edges, solve first SW-TVC independently **every possible  $P_3$**  in  $(G, \lambda)$
- take the **union** of the solutions

Lemma (Hamm, Klobas, Mertzios, Spirakis, AAI, 2022)

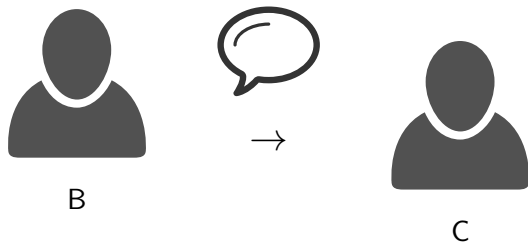
*There is a  $O(m^2T^2)$ -time  $(d - 1)$ -approximation algorithm for SW-TVC on **always degree at most  $d$**  temporal graphs.*

- We suspect an approximation ratio  $c \cdot d \dots$

- Temporal graphs
- Temporal paths: a warm-up
- Temporal vertex cover
- **Temporal transitive orientations**
- Stochastic temporal graphs

# Temporal transitive orientation

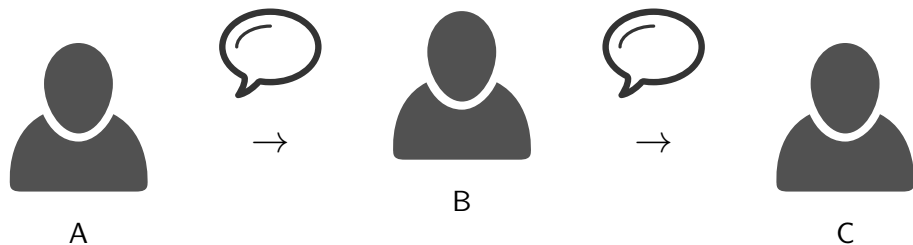
Motivation: Rumor Spreading



Scenario: C hears a rumor from B, asks for the source A, then (later) confirms with A whether the rumor is true.

# Temporal transitive orientation

Motivation: Rumor Spreading

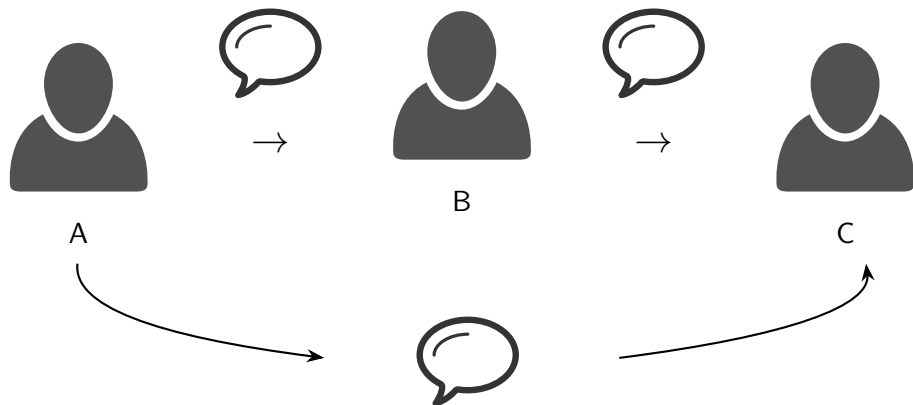


Scenario: C hears a rumor from B, asks for the source A, then (later) confirms with A whether the rumor is true.



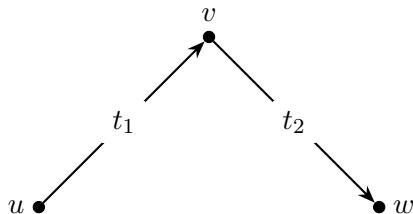
# Temporal transitive orientation

Motivation: Rumor Spreading



Scenario: C hears a rumor from B, asks for the source A, then (later) confirms with A whether the rumor is true.

# Temporal Transitivity

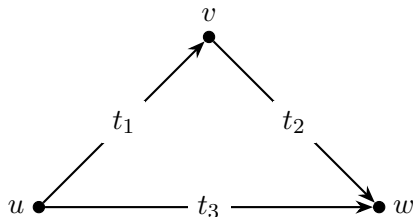


## Temporal Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  are temporal edges with  $t_1 \leq t_2$ ,

[Mertzios, Molter, Renken, Spirakis, Zschoche, *MFCS*, 2021]

# Temporal Transitivity

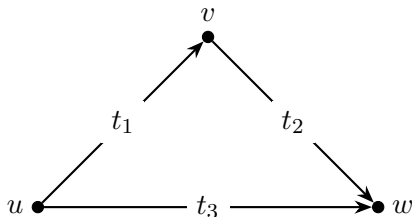


## Temporal Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  are temporal edges with  $t_1 \leq t_2$ , then  $(uw, t_3)$  is a temporal edge with  $t_2 \leq t_3$ .

[Mertzios, Molter, Renken, Spirakis, Zschoche, *MFCS*, 2021]

# Temporal Transitivity



## Temporal Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  are temporal edges with  $t_1 \leq t_2$ , then  $(uw, t_3)$  is a temporal edge with  $t_2 \leq t_3$ .

Exchanging  $\leq$  by  $<$  yields four variants:

### Temporal $(\{<, \leq\}, \{<, \leq\})$ -Transitivity

- first “ $<$ ” is called “strict”; second “ $<$ ” is called “strong”

[Mertzios, Molter, Renken, Spirakis, Zschoche, *MFCS*, 2021]

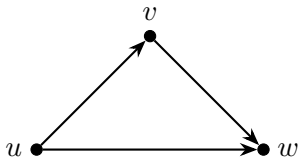
## Definition

A graph is **transitively orientable** if its edges can be oriented such that, if  $uv$  and  $vw$  are oriented edges, then  $uw$  exists in the graph and is an oriented edge.

# Static Transitivity

## Definition

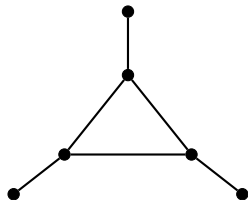
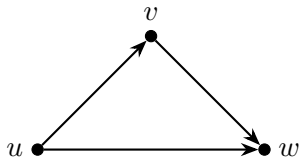
A graph is **transitively orientable** if its edges can be oriented such that, if  $uv$  and  $vw$  are oriented edges, then  $uw$  exists in the graph and is an oriented edge.



# Static Transitivity

## Definition

A graph is **transitively orientable** if its edges can be oriented such that, if  $uv$  and  $vw$  are oriented edges, then  $uw$  exists in the graph and is an oriented edge.

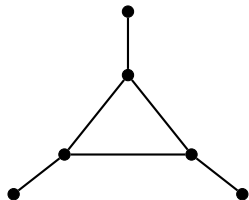
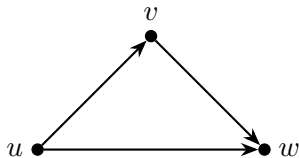


Forbidden induced subgraph

# Static Transitivity

## Definition

A graph is **transitively orientable** if its edges can be oriented such that, if  $uv$  and  $vw$  are oriented edges, then  $uw$  exists in the graph and is an oriented edge.



Forbidden induced subgraph

Transitively orientable graphs can be recognized in polynomial time [see.g. Golumbic '80].



# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

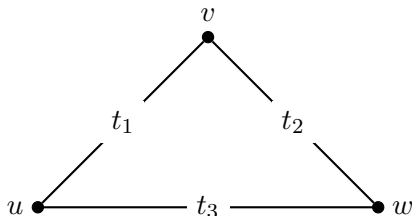
If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .

# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .

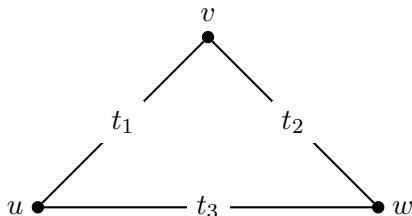


# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .



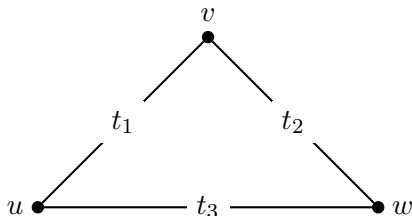
$$t_1 = t_2 = t_3 \mid t_1 < t_2 = t_3 \mid t_1 = t_2 < t_3 \mid t_1 < t_2 < t_3$$

# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .



$$t_1 = t_2 = t_3 \mid t_1 < t_2 = t_3 \mid t_1 = t_2 < t_3 \mid t_1 < t_2 < t_3$$

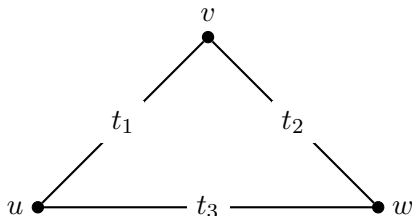
non-cyclic

# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .



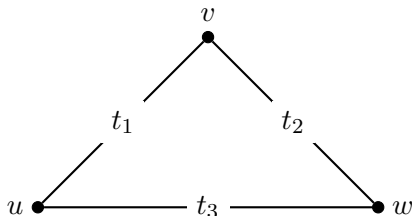
$$\begin{array}{l|l} t_1 = t_2 = t_3 & t_1 < t_2 = t_3 \quad | \quad t_1 = t_2 < t_3 \quad | \quad t_1 < t_2 < t_3 \\ \text{non-cyclic} & wu = vw \end{array}$$

# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

## Temporal $(\leq, \leq)$ -Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .



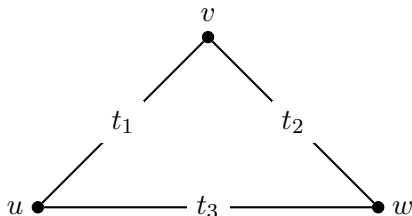
$t_1 = t_2 = t_3$	$t_1 < t_2 = t_3$	$t_1 = t_2 < t_3$	$t_1 < t_2 < t_3$
non-cyclic	$wu = wv$	$wv \implies uw$	$wu \implies wv$

# Recognizing Temporal Transitivity I

- We assume for simplicity **exactly one** temporal label per edge

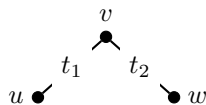
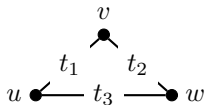
## Temporal ( $\leq, \leq$ )-Transitivity

If  $(uv, t_1)$  and  $(vw, t_2)$  with  $t_1 \leq t_2$ , then  $(uw, t_3)$  with  $t_2 \leq t_3$ .



$t_1 = t_2 = t_3$	$t_1 < t_2 = t_3$	$t_1 = t_2 < t_3$	$t_1 < t_2 < t_3$
non-cyclic	$wu = wv$	$vw \implies uw$ $vu \implies wu$	$vw \implies uw$ $vu \implies wu$

# Recognizing Temporal Transitivity II



$t_1 = t_2 = t_3$

$t_1 < t_2 = t_3$

$t_1 \leq t_2 < t_3$

$t_1 = t_2$

$t_1 < t_2$

$(\leq, \leq)$

**non-cyclic**

$wu = vw$

$vw \implies uw$

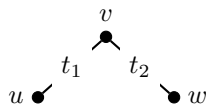
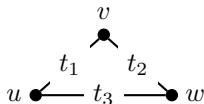
$vu \implies wu$

$uv = vw$

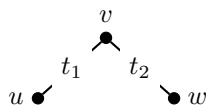
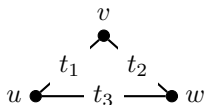
$uv \implies vw$



# Recognizing Temporal Transitivity II

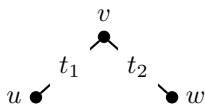
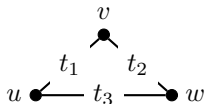

 $t_1 = t_2 = t_3$ 
 $t_1 < t_2 = t_3$ 
 $t_1 \leq t_2 < t_3$ 
 $t_1 = t_2$ 
 $t_1 < t_2$ 
 $(\leq, \leq)$ 
**non-cyclic**
 $wu = wv$ 
 $vw \implies uw$ 
 $vu \implies wu$ 
 $uv = wv$ 
 $uv \implies wv$ 
 $(\leq, <)$ 
 $\perp$ 
 $wu \wedge wv$ 
 $vw \implies uw$ 
 $vu \implies wu$ 
 $uv = wv$ 
 $uv \implies wv$

# Recognizing Temporal Transitivity II



	$t_1 = t_2 = t_3$	$t_1 < t_2 = t_3$	$t_1 \leq t_2 < t_3$	$t_1 = t_2$	$t_1 < t_2$
$(\leq, \leq)$	<b>non-cyclic</b>	$wu = wv$	$vw \implies uw$ $vu \implies wu$	$uv = wv$	$uv \implies wv$
$(\leq, <)$	$\perp$	$wu \wedge wv$	$vw \implies uw$ $vu \implies wu$	$uv = wv$	$uv \implies wv$
$(<, \leq)$	$\top$	<b>non-cyclic</b>	$vw \implies uw$ $vu \implies wu$	$\top$	$uv \implies wv$

# Recognizing Temporal Transitivity II



	$t_1 = t_2 = t_3$	$t_1 < t_2 = t_3$	$t_1 \leq t_2 < t_3$	$t_1 = t_2$	$t_1 < t_2$
$(\leq, \leq)$	<b>non-cyclic</b>	$wu = wv$	$vw \implies uw$ $vu \implies wu$	$uv = wv$	$uv \implies wv$
$(\leq, <)$	$\perp$	$wu \wedge wv$	$vw \implies uw$ $vu \implies wu$	$uv = wv$	$uv \implies wv$
$(<, \leq)$	$\top$	<b>non-cyclic</b>	$vw \implies uw$ $vu \implies wu$	$\top$	$uv \implies wv$
$(<, <)$	$\top$	$wu \wedge wv$	$vw \implies uw$ $vu \implies wu$	$\top$	$uv \implies wv$

## Recognizing Temporal Transitivity

- Recognizing of **Non-Strict** Temporal  $(\leq, \leq)$ -Transitivity in **poly-time**.
- Recognizing **Strict** Temporal  $(<, \leq)$ -Transitivity is **NP-hard**.
- Remaining (“strong”) variants can be recognized in **polynomial time**.

## Temporal Transitivity Completion

(given a partially oriented graph, add  $\leq k$  edges, and one label per edge)

- All four variants are NP-hard.
- Poly-time if input graph is fully oriented.
- FPT wrt. number of unoriented edges in input graph.

## Recognizing Multilayer Transitivity

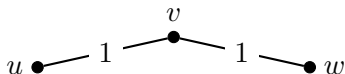
(permanent orientation of edges in a temporal graph)

- NP-hard.

[Mertzios, Molter, Renken, Spirakis, Zschoche, *MFCS*, 2021]

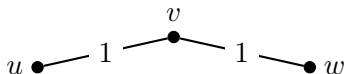
# Poly-time Algorithm for Temporal $(\leq, \leq)$ -Transitivity I

Important concept: **Forcing**:



# Poly-time Algorithm for Temporal ( $\leq, \leq$ )-Transitivity I

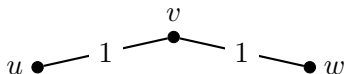
Important concept: **Forcing**:



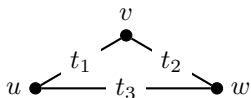
Main idea: Create a mixed Boolean formula  $\phi_{3NAE} \wedge \phi_{2SAT}$  from:

# Poly-time Algorithm for Temporal ( $\leq, \leq$ )-Transitivity I

Important concept: **Forcing**:



Main idea: Create a mixed Boolean formula  $\phi_{3NAE} \wedge \phi_{2SAT}$  from:

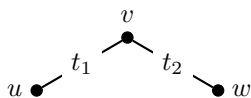


$t_1 = t_2 = t_3$      $t_1 < t_2 = t_3$      $t_1 \leq t_2 < t_3$

non-cyclic

$wu = wv$

$vw \implies uw$   
 $vu \implies wu$

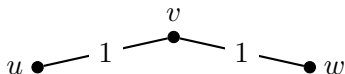


$t_1 = t_2$      $t_1 < t_2$

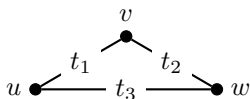
$uv = wv$      $uv \implies wv$

# Poly-time Algorithm for Temporal ( $\leq, \leq$ )-Transitivity I

Important concept: **Forcing**:



Main idea: Create a mixed Boolean formula  $\phi_{3NAE} \wedge \phi_{2SAT}$  from:

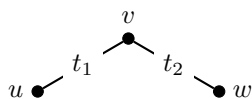


$t_1 = t_2 = t_3$      $t_1 < t_2 = t_3$      $t_1 \leq t_2 < t_3$

non-cyclic

$wu = wv$

$vw \implies uw$   
 $vu \implies wu$



$t_1 = t_2$      $t_1 < t_2$

$uv = wv$      $uv \implies wv$

Similar algorithm with solving 2SAT:

- 1 Set a variable, apply all “static forcings”: if no contradiction, keep it.
- 2 Iteratively set truth values and replace  $\phi_{3NAE}$ -clauses with  $\phi_{2SAT}$ -clauses.



# Poly-time Algorithm for Temporal $(\leq, \leq)$ -Transitivity II

Some key insights:

## Lemma

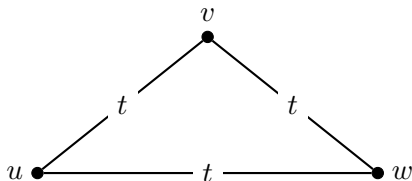
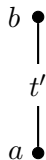
If orienting an edge forces orienting an edge in a “synchronous triangle”, it also forces orienting a **different** edge in the **same** synchronous triangle.

# Poly-time Algorithm for Temporal $(\leq, \leq)$ -Transitivity II

Some key insights:

## Lemma

If orienting an edge forces orienting an edge in a “synchronous triangle”, it also forces orienting a **different** edge in the **same** synchronous triangle.

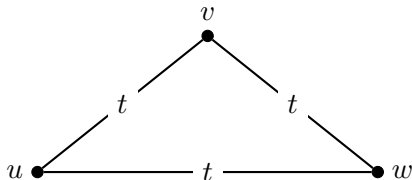
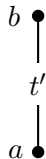


# Poly-time Algorithm for Temporal ( $\leq, \leq$ )-Transitivity II

Some key insights:

## Lemma

If orienting an edge forces orienting an edge in a “synchronous triangle”, it also forces orienting a **different** edge in the **same** synchronous triangle.

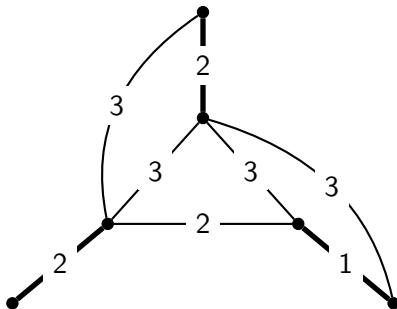


## Lemma

Transforming NAE-clauses into 2SAT-clauses creates **no** “new implication chains”.

# NP-hardness of (Strict) Temporal ( $<$ , $\leq$ )-Transitivity

Reduction from 3SAT. Clause gadget:



## Observation

Not all three thick edges can be oriented inwards, two inwards and one outwards possible.

- Temporal graphs
- Temporal parameters and temporal paths: a warm-up
- Temporal vertex cover
- Temporal transitive orientations
- Stochastic temporal graphs

# Stochastic Temporal Graphs

Levels of **knowledge** about the **network evolution**:

- whole temporal graph given in **advance**
- **adversary** who reveals it snapshot-by-snapshot at every time step
- **intermediate knowledge setting**, captured by **stochastic temporal graphs**, where the network evolution is given by a **probability distribution** that governs the appearance of each edge over time

[Akrida, Mertzios, Nikolettseas, Raptopoulos, Spirakis, Zamaraev,  
*J. Computer and System Sciences*, 2020]

# Stochastic Temporal Graphs

Levels of **knowledge** about the **network evolution**:

- whole temporal graph given in **advance**
- **adversary** who reveals it snapshot-by-snapshot at every time step
- **intermediate knowledge setting**, captured by **stochastic temporal graphs**, where the network evolution is given by a **probability distribution** that governs the appearance of each edge over time

“Memory effect”: appearance **probability** of a particular edge at a given time **step  $t$**  depends on the appearance (or absence) of the same edge at the **previous  $k \geq 1$  time steps**

- faulty network communication

[Akrida, Mertzios, Nikolettseas, Raptopoulos, Spirakis, Zamaraev, *J. Computer and System Sciences*, 2020]

# Stochastic Temporal Graphs

**Memoryless case,  $\mathcal{G}^{(0)}$ :**

$\forall e \in E, \forall t \in \mathbb{N}$ ,  $e$  appears in  $G_t$  with probability  $p_e$ .

The numbers  $\{p_e : e \in E\}$  are given parameters of the model.



# Stochastic Temporal Graphs

**Memoryless case,  $\mathcal{G}^{(0)}$ :**

$\forall e \in E, \forall t \in \mathbb{N}$ ,  $e$  appears in  $G_t$  with probability  $p_e$ .

The numbers  $\{p_e : e \in E\}$  are given parameters of the model.

**Memory-1,  $\mathcal{G}^{(1)}$ :**

**Initial snapshot**  $G_0 \subseteq G$ .

$\forall e \in E, \forall t \in \mathbb{N}$ :

- if  $e$  was absent in  $G_{t-1}$ ,  $e$  appears in  $G_t$  with probability  $p_e$  and is absent with probability  $1 - p_e$
- if  $e$  appeared in  $G_{t-1}$ ,  $e$  appears in  $G_t$  with probability  $1 - q_e$  and is absent with probability  $q_e$

$$M_e = \left( \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 1 - p_e & p_e \\ 1 & q_e & 1 - q_e \end{array} \right), \text{ where } 0 \leq p_e, q_e \leq 1.$$

# Stochastic Temporal Graphs

## Memoryless case, $\mathcal{G}^{(0)}$ :

$\forall e \in E, \forall t \in \mathbb{N}$ ,  $e$  appears in  $G_t$  with probability  $p_e$ .

The numbers  $\{p_e : e \in E\}$  are given parameters of the model.

## Memory-1, $\mathcal{G}^{(1)}$ :

Initial snapshot  $G_0 \subseteq G$ .

$\forall e \in E, \forall t \in \mathbb{N}$ :

- if  $e$  was absent in  $G_{t-1}$ ,  $e$  appears in  $G_t$  with probability  $p_e$  and is absent with probability  $1 - p_e$
- if  $e$  appeared in  $G_{t-1}$ ,  $e$  appears in  $G_t$  with probability  $1 - q_e$  and is absent with probability  $q_e$

$$M_e = \left( \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 1 - p_e & p_e \\ 1 & q_e & 1 - q_e \end{array} \right), \text{ where } 0 \leq p_e, q_e \leq 1.$$

If  $p_e = p$  and  $q_e = q$ ,  $\forall e$ , we have exactly the *edge-Markovian evolving graph* model introduced by Clementi et al. (SIAM Journal on Discrete Mathematics '10).

# Stochastic Temporal Graphs

## Memoryless case, $\mathcal{G}^{(0)}$ :

$\forall e \in E, \forall t \in \mathbb{N}$ ,  $e$  appears in  $G_t$  with probability  $p_e$ .

The numbers  $\{p_e : e \in E\}$  are given parameters of the model.

## Memory- $k$ , $\mathcal{G}^{(k)}$ :

Initial sequence of  $k$  snapshots  $G_{-k+1}, \dots, G_{-1}, G_0 \subseteq G$ .

$\forall e \in E, \forall t \in \mathbb{N}$ :

- $e$  appears with probability  $p_e(H_e^{(k)})$  that depends only on the history  $H_e^{(k)}$  of its appearance in the last  $k$  snapshots.
- at every time step  $t$ , this history is a  $k$ -bit binary vector, where a 0-entry (resp. 1-entry) on the  $i$ -th position denotes absence (resp. appearance) of  $e$  in  $E_{t-k+i-1}$ , for  $i = 1, \dots, k$

# Stochastic Temporal Graphs

## Memoryless case, $\mathcal{G}^{(0)}$ :

$\forall e \in E, \forall t \in \mathbb{N}$ ,  $e$  appears in  $G_t$  with probability  $p_e$ .

The numbers  $\{p_e : e \in E\}$  are given parameters of the model.

## Memory- $k$ , $\mathcal{G}^{(k)}$ :

Initial sequence of  $k$  snapshots  $G_{-k+1}, \dots, G_{-1}, G_0 \subseteq G$ .

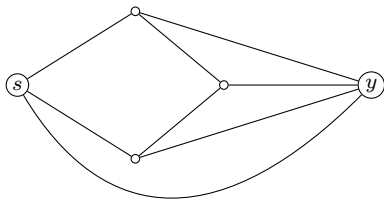
$\forall e \in E, \forall t \in \mathbb{N}$ :

- $e$  appears with probability  $p_e(H_e^{(k)})$  that depends only on the history  $H_e^{(k)}$  of its appearance in the last  $k$  snapshots.
- at every time step  $t$ , this history is a  $k$ -bit binary vector, where a 0-entry (resp. 1-entry) on the  $i$ -th position denotes absence (resp. appearance) of  $e$  in  $E_{t-k+i-1}$ , for  $i = 1, \dots, k$

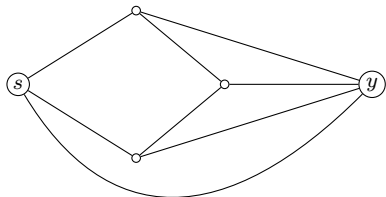
For every  $k \geq 1$ , the memory- $(k-1)$  model is a special case of the memory- $k$  model.

# The problems

**Unbounded** number of messages:  
“Flooding” the network with  
information

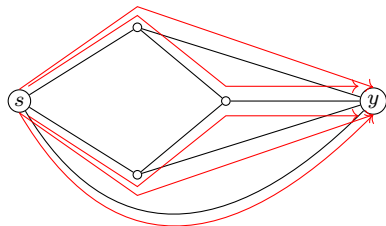


**Limited** number of messages:  
transferring a package with a  
tangible good

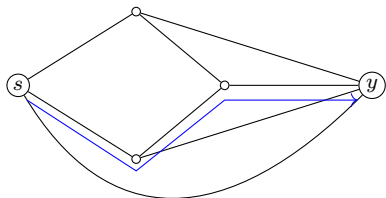


# The problems

**Unbounded** number of messages:  
“Flooding” the network with  
information



**Limited** number of messages:  
transferring a package with a  
tangible good

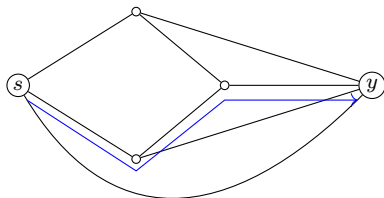


# The problems

## MINIMUM ARRIVAL:

Given a stochastic temporal graph on an underlying graph  $G = (V, E)$  and two distinct vertices  $s, y \in V$ , compute the **expected arrival time of a foremost  $s$ - $y$  journey**,  $\mathbb{E}[X(s, y)]$ .

Limited number of messages:  
transferring a package with a  
tangible good



## MINIMUM ARRIVAL:

Given a stochastic temporal graph on an underlying graph  $G = (V, E)$  and two distinct vertices  $s, y \in V$ , compute the **expected arrival time of a foremost  $s$ - $y$  journey**,  $\mathbb{E}[X(s, y)]$ .

## BEST POLICY:

Every day  $t$  Alice “wakes up” in the morning located at vertex  $s_t$  and looks at **which edges are available** in **today’s** snapshot; by only knowing her **current position**, the **history** of the last  $k$  snapshots, and the probabilistic **rules of edge appearance**, Alice needs to decide whether:

- to **stay** at the vertex  $s_t$  she currently is, or
- to use an edge of  $G_t$  to **move** to a neighbouring vertex.



## MINIMUM ARRIVAL:

Given a stochastic temporal graph on an underlying graph  $G = (V, E)$  and two distinct vertices  $s, y \in V$ , compute the **expected arrival time of a foremost  $s$ - $y$  journey**,  $\mathbb{E}[X(s, y)]$ .

## BEST POLICY:

Given a stochastic temporal graph on an underlying graph  $G = (V, E)$  and two distinct vertices  $s, y \in V$ , compute the **expected arrival time of a best policy  $s$ - $y$  journey**,  $\mathbb{E}[Y(s, y)]$ .

# The problems

## MINIMUM ARRIVAL:

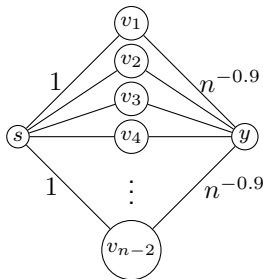
Arrival time of the foremost journey from  $s$  to  $y$  will be equal to the **first day after day 1** on which some edge incident to  $y$  appears.

Time needed for that follows geometric distribution, with success probability

$$1 - (1 - n^{-0.9})^{n-2} = 1 - o(1).$$

So, solution is:

$$E[X(s, y)] = 2 + o(1).$$



## BEST POLICY:

Any best policy for Alice will **cross an edge incident to  $s$  on day 1** and then **wait** until the “next” edge in the path, incident to  $y$ , appears.

Time needed for that to happen is  $n^{0.9}$ .

So, solution is:

$$E[Y(s, y)] = 1 + n^{0.9}.$$

## MINIMUM ARRIVAL:

- #P-hard (even for the memoryless case)
- Approximation Scheme for memory-0 on series-parallel graphs
- Fully Polynomial Randomized Approximation Scheme (FPRAS) for memory- $k$ ,  $k \geq 0$

## BEST POLICY:

- #P-hard for memory- $k$ ,  $k \geq 3$
- Formulation as MDP, leading to exact doubly-exponential-time algorithm
- Polynomial-time dynamic programming algorithm for the memoryless case
  - Studied before; different approaches; polynomial-time solutions e.g. [Ogier and Rutenburg, Infocom '92](#) & [Basu et al., arXiv](#)

- Parameterized versions of the problems (with the appropriate parameters)
- Approximation algorithms
- Special temporal graph classes
  - e.g. the class of temporally orientable temporal graphs...?
- Distinction for path problems: strict vs. non-strict
  - the same distinction also on derived notions, e.g. temporal transitivity
- Other meaningful temporal graph problems
  - lifting “algorithmic graph theory” to the temporal case
- Need for experimental algorithms
  - Experimental Algorithms are needed especially for the provably hard problems here

Thank you for your attention!