Algorithms and Complexity on Temporal Graphs

George B. Mertzios

School of Engineering and Computing Sciences,
Durham University, UK

Graduiertenkolleg – Monday Lecture:
Methods for Discrete Structures

TU Berlin / FU Berlin / HU Berlin

July 2016

Static and Temporal Graphs

Many systems in Science and Technology:

- abstracted as graphs
- vertex $\longleftrightarrow$ elementary system unit
- edge $\longleftrightarrow$ some kind of interaction between units

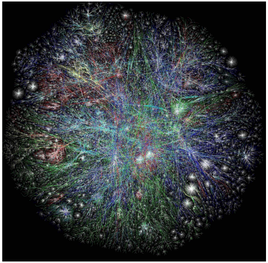
Static and Temporal Graphs

Many systems in Science and Technology:

- abstracted as graphs
- vertex $\longleftrightarrow$ elementary system unit
- edge $\longleftrightarrow$ some kind of interaction between units

However many modern systems are highly dynamic:

- Modern communication networks:
  - links change dynamically at a high rate
  - mobile ad hoc, sensor, peer-to-peer, opportunistic, delay-tolerant networks, etc.
- Network changes may:
  - follow specific patterns, e.g. satellites following a trajectory, or
  - be unpredictable, e.g. mobile ad hoc networks



The internet graph.

Static and Temporal Graphs

Further examples of modern dynamic systems:

- 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

Static and Temporal Graphs

Further examples of modern dynamic systems:

- 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

$\Rightarrow$ 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

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Temporal graphs

Formally:

Definition (Temporal Graph)

A temporal graph is a pair (G, λ) where:

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

Temporal graphs

Formally:

Definition (Temporal Graph)

A temporal graph is a pair (G, λ) where:

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

- If $t \in \lambda(e)$ then edge e is available at time t
- This formal definition (for single-availabilities per edge) embarks from:
  [Kempe, Kleinberg, Kumar, *STOC*, 2000]
  [Berman, *Networks*, 1996]
- In general every edge can have multiple availabilities
  [Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013]

Temporal graphs

Formally:

Definition (Temporal Graph)

A temporal graph is a pair (G, λ) where:

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

Remarks:

- Denote $\lambda_{\text{min}}/\lambda_{\text{max}}$ be the smallest/largest time-label in (G, λ)
- λ_{max} can also be infinite (e.g. in periodic temporal graphs)
- Otherwise the the age of (G, λ) is $\alpha(\lambda) = \lambda_{\text{max}} - \lambda_{\text{min}} + 1$
- Unless otherwise specified:
  - the labels are given explicitly with the input
  - $c(\lambda)$ is the total number of labels

Temporal graphs

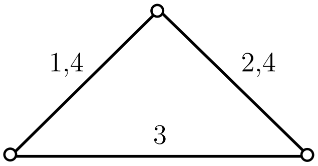
Formally:

Definition (Temporal Graph)

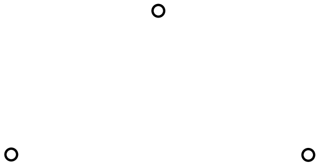
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

Formally:

Definition (Temporal Graph)

A temporal graph is a pair (G, λ) where:

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

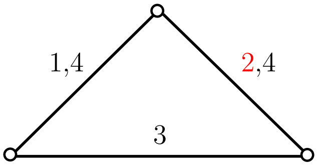
Formally:

Definition (Temporal Graph)

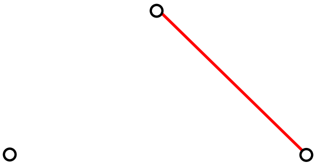
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

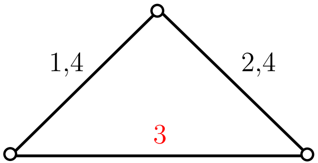
Formally:

Definition (Temporal Graph)

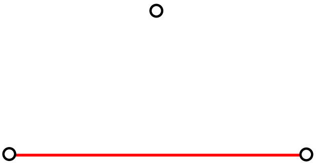
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

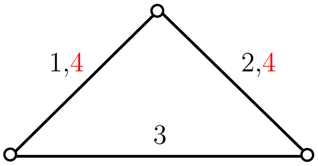
Formally:

Definition (Temporal Graph)

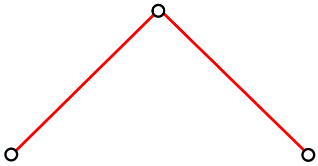
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

Related models

Related notions of dynamicity in graphs:

- flows over time
  [Fleischer, Skutella, *SIAM J. on Computing*, 2007]
  [Hoppe, Tardos, *Math. Oper. Res.*, 2000]
  [Fleischer, Tardos, *Oper. Res. Lett.*, 1998]
    - flows on static graph topologies with transit times on the edges
    - continuous availabilities; natural model, different techniques

Temporal graphs
Related models

Related notions of dynamicity in graphs:

- flows over time
  [Fleischer, Skutella, *SIAM J. on Computing*, 2007]
  [Hoppe, Tardos, *Math. Oper. Res.*, 2000]
  [Fleischer, Tardos, *Oper. Res. Lett.*, 1998]
  - flows on static graph topologies with transit times on the edges
  - continuous availabilities; natural model, different techniques
- minimum label graph problems
  [Fellows, Guo, Kanj, *J. Comp. Syst. Sci.*, 2010]
  - input: static topology G with a label on each edge, graph property Π
  - goal: find an edge subset with the smallest number of distinct labels which satisfies Π

Related notions of dynamicity in graphs:

- dynamic graphs
  [Demetrescu, Finocchi, Italiano, *Handbook Data Str. and Appl.*, 2004]
  - topology changes via insertion/deletion of vertices/edges
  - changes are assumed to happen rarely
  - goals: efficient query & solution update after a dynamic change

Related notions of dynamicity in graphs:

- dynamic graphs
  [Demetrescu, Finocchi, Italiano, *Handbook Data Str. and Appl.*, 2004]
  - topology changes via insertion/deletion of vertices/edges
  - changes are assumed to happen rarely
  - goals: efficient query & solution update after a dynamic change

In contrast, in context of temporal networks:

- topology is expected to change frequently and massively

$\Rightarrow$ changes are not anomalies or exceptions

- they are rather an integral part of the system

$\Rightarrow$ can not be reasonably modeled with network faults /failures

Temporal graphs

Temporal graphs were studied under various different names:

- time-varying graphs
  [Aaron et al., *WG*, 2014]
  [Flocchini et al., *ISAAC*, 2009]
  [Tang et al., *ACM Comp. Comm. Review*, 2010]
- evolving graphs (usually "graph-centric")
  [Avin et al., *ICALP*, 2008]
  [Clementi et al., *SIAM J. Discr. Math.*, 2010]
  [Ferreira, *IEEE Network*, 2004]
  [Bui Xuan et al., *Int. J. Found. Comp. Sci.*, 2003]
- dynamic graphs
  [Giakkoupis et al., *ICALP*, 2014]
  [Casteigts et al., *Int. J. Par., Emergent & Distr. Syst*, 2012]
  [Bhadra and Ferreira, *ADHOC-NOW*, 2003]
- graphs over time
  [Leskovec et al., *ACM Trans. Knowl. Disc. from Data*, 2007]

Temporal graphs

Recent surveys and books:

- Time-Varying Graphs and Dynamic Networks
  [Casteigts et al., *Int. J. Par., Emergent & Distr. Syst*, 2012]
    - an attempt to integrate and unify existing models and concepts

- Deterministic Algorithms in Dynamic Networks
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report I*, 2013]
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report II*, 2013]
    - survey of deterministic algorithms for distributed computing
    - temporal graph classes based on temporal patterns of the labels

Temporal graphs

Recent surveys and books:

- Time-Varying Graphs and Dynamic Networks
  [Casteigts et al., *Int. J. Par., Emergent & Distr. Syst*, 2012]
  - an attempt to integrate and unify existing models and concepts

- Deterministic Algorithms in Dynamic Networks
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report I*, 2013]
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report II*, 2013]
  - survey of deterministic algorithms for distributed computing
  - temporal graph classes based on temporal patterns of the labels

    – satellites $\longrightarrow$ periodic availabilities
    – sensor networks $\longrightarrow$ connected at every instant
    – contacts in a company $\longrightarrow$ bounded edge recurrence (every week)
    – community contacts $\longrightarrow$ unbounded, yet recurrent interactions

Temporal graphs

Recent surveys and books:

- Time-Varying Graphs and Dynamic Networks
  [Casteigts et al., *Int. J. Par., Emergent & Distr. Syst*, 2012]
  - an attempt to integrate and unify existing models and concepts
- Deterministic Algorithms in Dynamic Networks
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report I*, 2013]
  [Casteigts, Flocchini, *Defence R&D Canada, Tech. Report II*, 2013]
  - survey of deterministic algorithms for distributed computing
  - temporal graph classes based on temporal patterns of the labels
    - satellites $\longrightarrow$ periodic availabilities
    - sensor networks $\longrightarrow$ connected at every instant
    - contacts in a company $\longrightarrow$ bounded edge recurrence (every week)
    - community contacts $\longrightarrow$ unbounded, yet recurrent interactions
- Temporal Networks [Holme, Saramäki, eds., *Springer*, 2013]
  - temporal network methods for complex networks

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

$\Rightarrow$ strictly increasing labels along the path

- a "static path" given "in pieces"

Most identified temporal graph parameters are "temporal path"-related:

- temporal versions of distance, diameter, connectivity, reachability, exploration, centrality measures, etc.

Temporal paths

The most natural known temporal notion in temporal graphs:

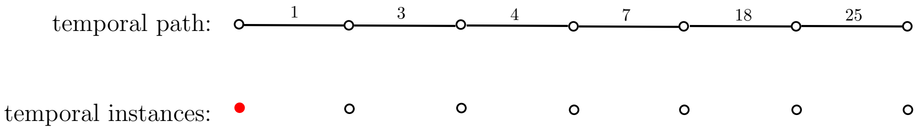
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

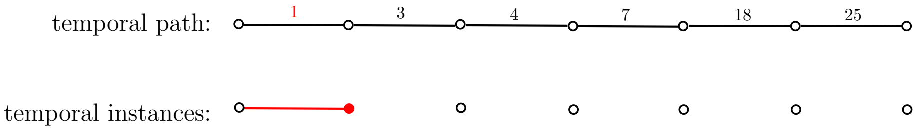
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

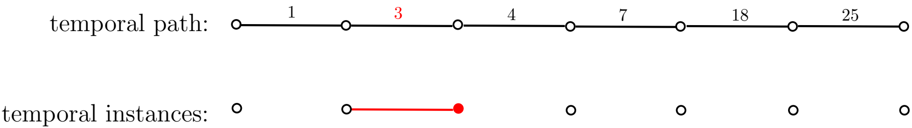
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

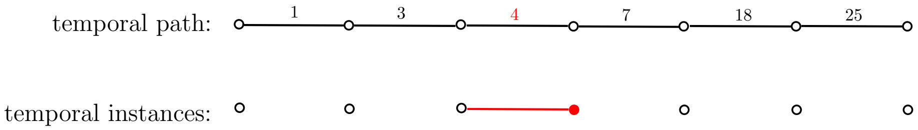
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

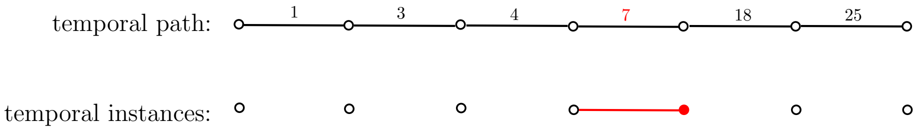
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

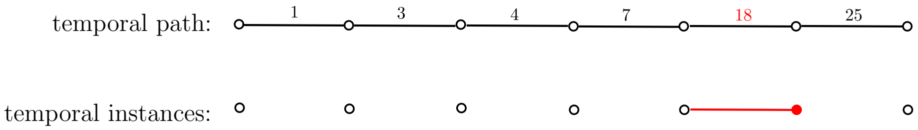
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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:

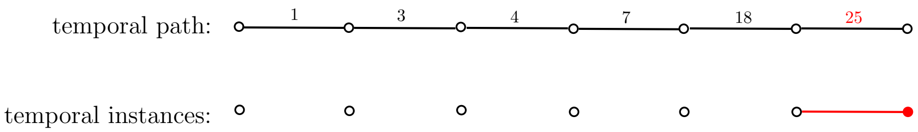
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

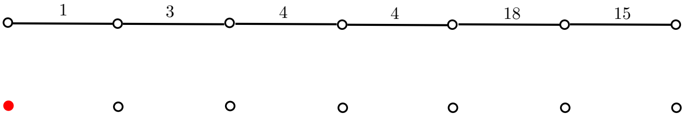
$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:

non-temporal path:

temporal instances:



Temporal paths

The most natural known temporal notion in temporal graphs:

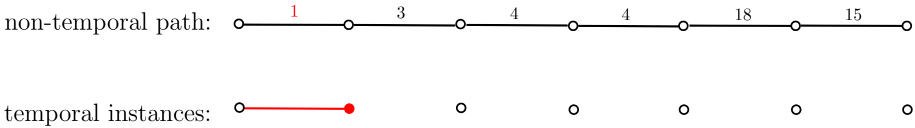
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:



Temporal paths

The most natural known temporal notion in temporal graphs:

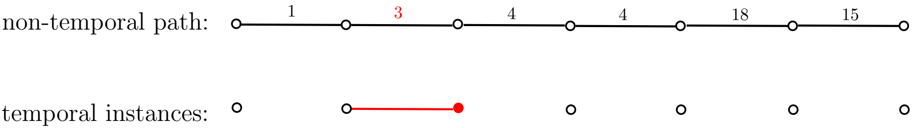
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:



Temporal paths

The most natural known temporal notion in temporal graphs:

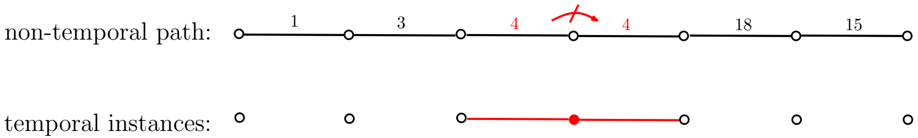
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:



Temporal paths

The most natural known temporal notion in temporal graphs:

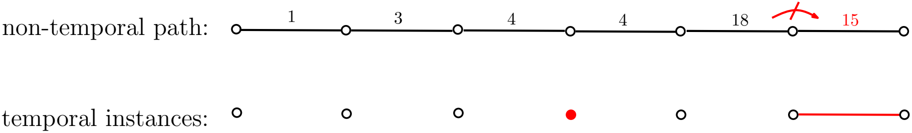
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:



Temporal paths

The most natural known temporal notion in temporal graphs:

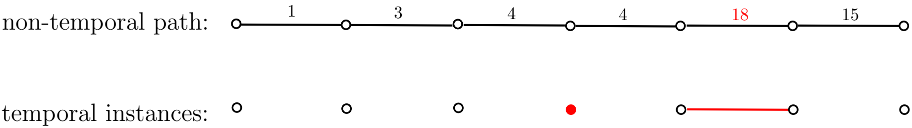
Definition (Temporal path; Time-respecting path; Journey)

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

$$\ell_1 < \ell_2 < \ldots < \ell_k$$

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

- A non-temporal path:



Metrics to optimize

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

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

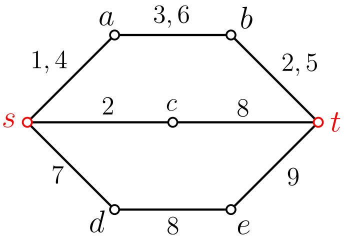
Metrics to optimize

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:



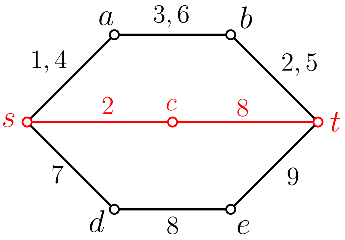
Metrics to optimize

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)

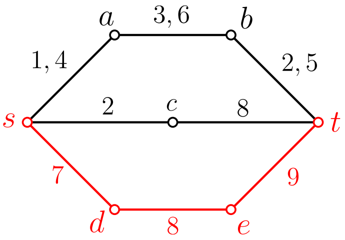
Metrics to optimize

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)

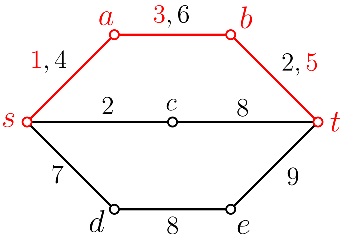
Metrics to optimize

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)

Metrics to optimize

An easy algorithm for computing all foremost paths from a given source s:
[Akrida, Gasieniec, Mertzios, Spirakis, *WAOA*, 2015]

- first sort the time-labels non-decreasingly
- run a BFS-like search starting from s
- at every time-step t consider only edges currently available
- if you reach a new vertex at time t, keep its predecessor

An easy algorithm for computing all foremost paths from a given source s:
[Akrida, Gasieniec, Mertzios, Spirakis, *WAOA*, 2015]

Algorithm 1 Foremost Temporal Paths from Source s

```
Let S be the array with the sorted time-labels
R ← {s}
for each v ∈ V \ {s} do
    pred[v] ← ∅;  arr[v] ← ∞        {Init.: Predecessor; Time Arrived}
for each time-label t ∈ S do
    for each edge e = (u, v) with t ∈ λ(e) do
        if u ∈ R,  v ∉ R,  and arr[u] < t then {we reached v}
            pred[v] ← u;  arr[v] ← t     {Predecessor; Time Arrived}
            R ← R ∪ {v}
```

Metrics to optimize

An easy algorithm for computing all foremost paths from a given source s:

- easy adaptation of the static BFS algorithm
- running time $O(c(\lambda) \cdot \log(c(\lambda)))$
- due to the sorting of the labels

Algorithm 1 Foremost Temporal Paths from Source s

Let S be the array with the sorted time-labels
$R \leftarrow \{s\}$
for each $v \in V \setminus \{s\}$ **do**
    $pred[v] \leftarrow \emptyset$; $arr[v] \leftarrow \infty$ {Init.: Predecessor; Time Arrived}
for each time-label $t \in S$ **do**
    for each edge $e = (u, v)$ with $t \in \lambda(e)$ **do**
        if $u \in R$, $v \notin R$, and $arr[u] < t$ **then** {we reached v}
            $pred[v] \leftarrow u$; $arr[v] \leftarrow t$ {Predecessor; Time Arrived}
            $R \leftarrow R \cup \{v\}$

Metrics to optimize

Polynomial algorithms exist also in the case of edges with traversal times for computing:

- shortest and foremost paths [adaptations of Dijkstra's algorithm]
- fastest paths

[Bui-Xuan, Ferreira, Jarry, *Int. J. Found. Comp. Sci.*, 2003]

Metrics to optimize

Polynomial algorithms exist also in the case of edges with traversal times for computing:

- shortest and foremost paths [adaptations of Dijkstra's algorithm]
- fastest paths

[Bui-Xuan, Ferreira, Jarry, *Int. J. Found. Comp. Sci.*, 2003]

Question: Are all "path-related" temporal problems tractable?

Metrics to optimize

Polynomial algorithms exist also in the case of edges with traversal times for computing:

- shortest and foremost paths [adaptations of Dijkstra's algorithm]
- fastest paths

[Bui-Xuan, Ferreira, Jarry, *Int. J. Found. Comp. Sci.*, 2003]

Question: Are all "path-related" temporal problems tractable?

Answer: Not all!

E.g. some temporal variations of:

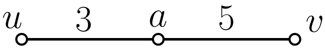
- connectivity problems
- reachability problems

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

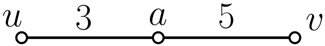
Temporal strongly connected components

- We write $u \rightsquigarrow v$ if there exists a temporal path from u to v
- The relation $\rightsquigarrow$ is not symmetric: $u \rightsquigarrow v \quad \not\Leftrightarrow \quad v \rightsquigarrow u$

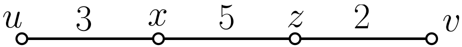


Temporal strongly connected components

- We write $u \rightsquigarrow v$ if there exists a temporal path from u to v
- The relation $\rightsquigarrow$ is not symmetric: $u \rightsquigarrow v \;\not\Leftrightarrow\; v \rightsquigarrow u$



- and not transitive: $u \rightsquigarrow z,\; z \rightsquigarrow v \;\not\Leftrightarrow\; u \rightsquigarrow v$



$\Rightarrow$ the time dimension creates its own "level of direction"

Temporal strongly connected components

Recall:

Definition

A directed (static) graph G is strongly connected if there is a path in each direction between each pair of vertices of G.

Temporal strongly connected components

Recall:

Definition

A directed (static) graph G is strongly connected if there is a path in each direction between each pair of vertices of G.

A key property:

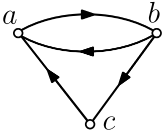
Observation

Let S be a (maximal) strongly connected subgraph and $u, v \in S$.
If $P = (u, \ldots, z, \ldots, v)$ is a path from u to v then $z \in S$.

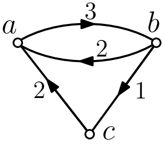
- Does this transfer to temporal graphs?

Temporal strongly connected components

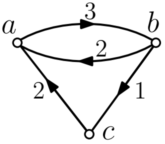
static:



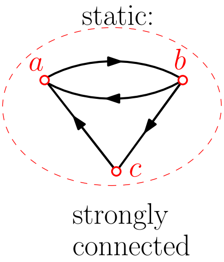
temporal:

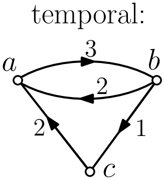


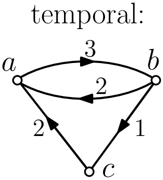
temporal:



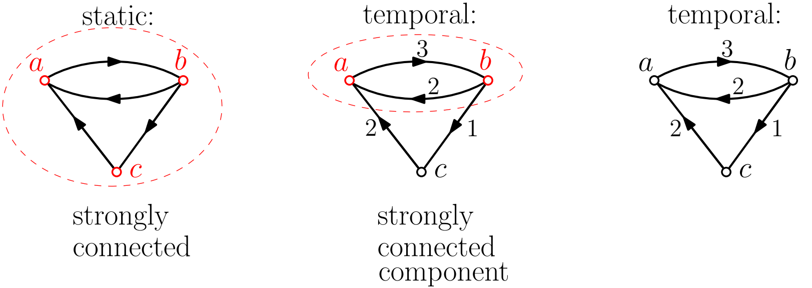
Temporal strongly connected components





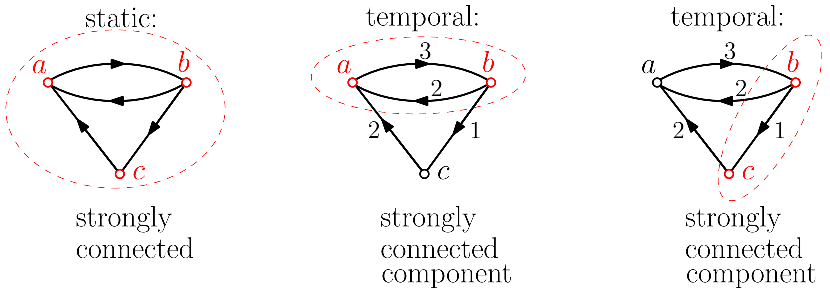


Temporal strongly connected components



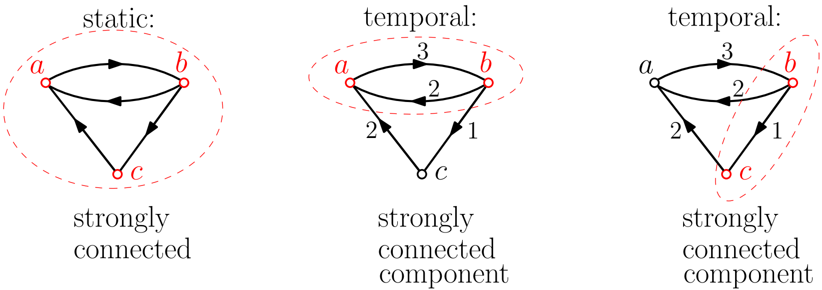
- $\{a, b\}$: direct temporal paths between a and b

Temporal strongly connected components



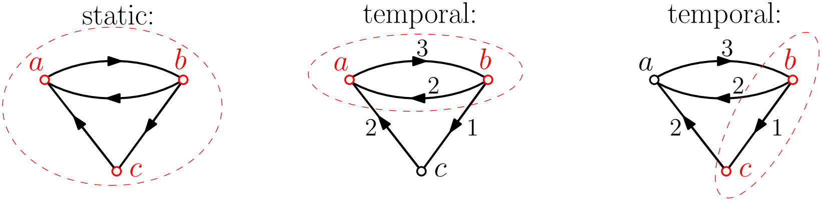
- $\{a, b\}$: direct temporal paths between a and b
- $\{b, c\}$: the only temporal path from c to b passes through $a \notin \{b, c\}$

Temporal strongly connected components



- $\{a, b\}$: direct temporal paths between a and b
- $\{b, c\}$: the only temporal path from c to b passes through $a \notin \{b, c\}$
- $\{a, b, c\}$: no temporal path from a to c

Temporal strongly connected components

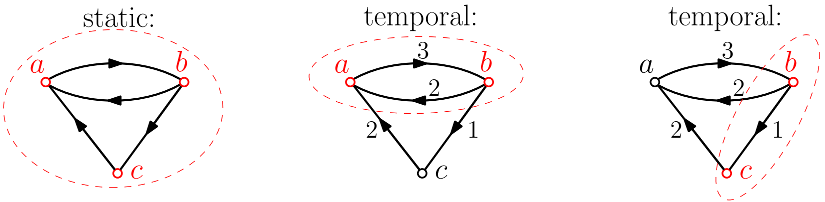


Definition (Bharda, Ferreira, 2003)

An open strongly connected component (o-SCC) in a temporal graph is a set S of vertices such that $u \rightsquigarrow v$ for every $u, v \in S$.

Examples of an o-SCC: $\{a, b\}$, $\{b, c\}$

Temporal strongly connected components



Definition (Bharda, Ferreira, 2003)

An open strongly connected component (o-SCC) in a temporal graph is a set S of vertices such that $u \rightsquigarrow v$ for every $u, v \in S$.

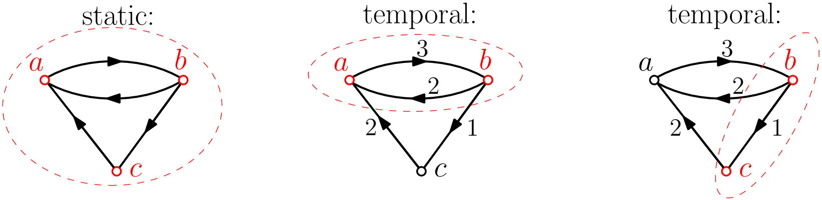
Examples of an o-SCC: $\{a, b\}$, $\{b, c\}$

Definition (Bharda, Ferreira, 2003)

A strongly connected component (SCC) in a temporal graph is a set S of vertices such that, for every $u, v \in S$, there is a temporal path from u to v that uses only vertices from S.

Example of a SCC: $\{a, b\}$

Temporal strongly connected components



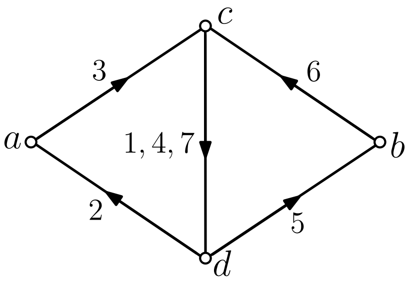
A difference to the static case:

- there can be a path between two vertices of the SCC (e.g. $\{a, b\}$) that traverses vertices outside the SCC (e.g. c)
- the same for an o-SCC (e.g. $\{b, c\}$)

Temporal strongly connected components

Further differences to the static case:

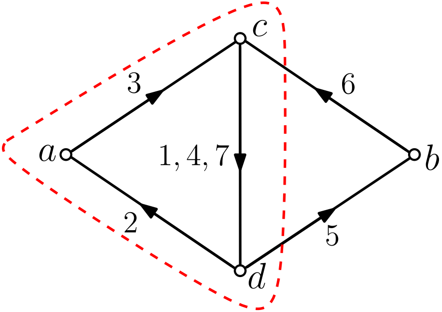
- two different SCCs can have common vertices



Temporal strongly connected components

Further differences to the static case:

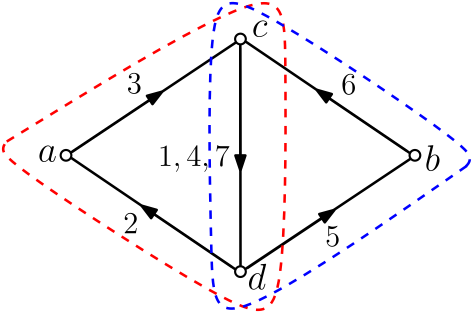
- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC



Temporal strongly connected components

Further differences to the static case:

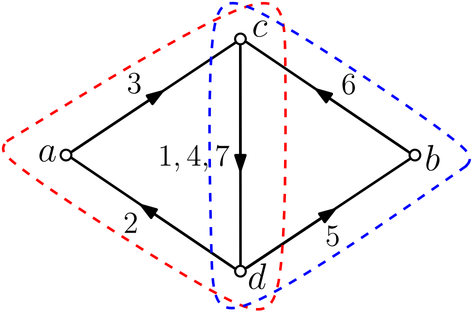
- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC
  - $\{b, c, d\}$ is another SCC



Temporal strongly connected components

Further differences to the static case:

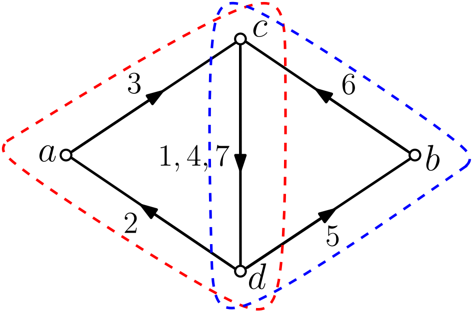
- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC
  - $\{b, c, d\}$ is another SCC
  - $\{a, b, c, d\}$ is not a SCC (no temporal path $b \rightsquigarrow a$)



Temporal strongly connected components

Further differences to the static case:

- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC
  - $\{b, c, d\}$ is another SCC
  - $\{a, b, c, d\}$ is not a SCC (no temporal path $b \rightsquigarrow a$)



- Can we compute/verify temporal SCCs/o-SCCs efficiently?

Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a vertex subset S of a temporal graph (G, λ), we can verify in polynomial time whether S is a SCC (resp. an o-SCC).

Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a vertex subset S of a temporal graph (G, λ), we can verify in polynomial time whether S is a SCC (resp. an o-SCC).

Proof.

- consider the induced (temporal) subgraph on S (resp. whole (G, λ))
- from every vertex $v \in S$ compute all foremost temporal paths
  - or all shortest / fastest paths, with any of the known algorithms
- if at least one vertex v does not reach the whole S (resp. whole G):
  - then S is not a SCC (resp. an o-SCC)

$\square$

Observation: similarly to static graphs

Temporal strongly connected components

Theorem

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of a SCC, even if all edges have one and the same label.

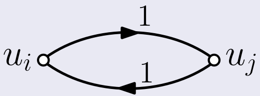
Temporal strongly connected components

Theorem

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of a SCC, even if all edges have one and the same label.

Proof.

- Reduction from CLIQUE.
- Given a static undirected graph G construct a temporal graph $(\mathcal{G}, \lambda)$:
  1. for each v_i of G create vertex u_i in $\mathcal{G}$,
  2. for each edge (v_i, v_j) of G add these two arcs to $\mathcal{G}$:



- G has a clique of size k $\Leftrightarrow$ $(\mathcal{G}, \lambda)$ has an SCC of size k.

$\square$

Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of an o-SCC, even if all edges have two labels.

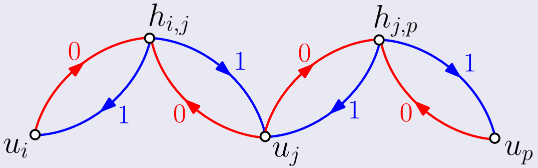
Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of an o-SCC, even if all edges have two labels.

Proof.

- Again reduction from CLIQUE.
- Given a static undirected graph G construct a temporal graph $(\mathcal{G}, \lambda)$:
    1. for each v_i of G create vertex u_i in $\mathcal{G}$,
    2. for each edge (v_i, v_j) of G create a vertex h_{ij} in $\mathcal{G}$ and:
       – add the arcs $(u_i, h_{ij}), (u_j, h_{ij})$ with label 0,
       – add the arcs $(h_{ij}, u_i), (h_{ij}, u_j)$ with label 1.

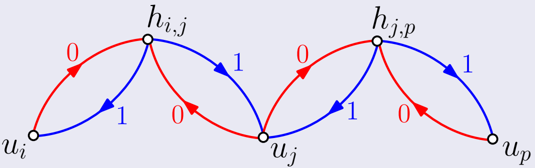


Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of an o-SCC, even if all edges have two labels.

Proof (continued).



- Whenever (v_i, v_j) is an edge in $G \Rightarrow u_i \rightsquigarrow u_j$ in $\mathcal{G}$
- A vertex h_{ij} has a temporal path only to u_i and u_j

$\Rightarrow$ any o-SCC with more than 3 vertices contains only u_i-vertices

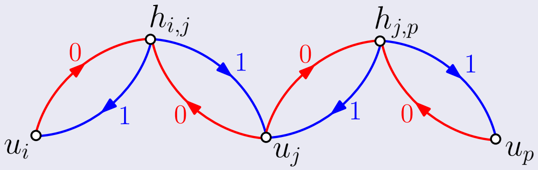
$\square$

Temporal strongly connected components

Theorem (Bharda, Ferreira, 2003)

Given a temporal graph (G, λ), it is NP-hard to compute the maximum size of an o-SCC, even if all edges have two labels.

Proof (continued).



- Whenever (v_i, v_j) is an edge in $G \Rightarrow u_i \rightsquigarrow u_j$ in $\mathcal{G}$
- A vertex h_{ij} has a temporal path only to u_i and u_j

$\Rightarrow$ any o-SCC with more than 3 vertices contains only u_i-vertices

$\Rightarrow$ G has a clique of size $k \Leftrightarrow (\mathcal{G}, \lambda)$ has an o-SCC of size k. □

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Menger's Theorem

Two fundamental duality results in (static) graph theory:

Theorem (Menger, 1927)

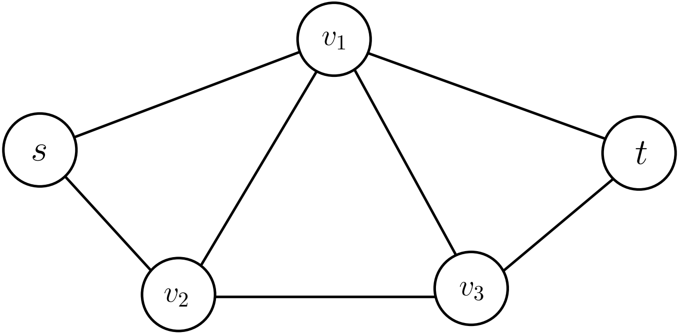
The maximum number of vertex-disjoint (edge-disjoint) s-t paths is equal to the minimum number of vertices (edges) needed to separate s from t.

Menger's Theorem

Two fundamental duality results in (static) graph theory:

Theorem (Menger, 1927)

The maximum number of vertex-disjoint (edge-disjoint) s-t paths is equal to the minimum number of vertices (edges) needed to separate s from t.

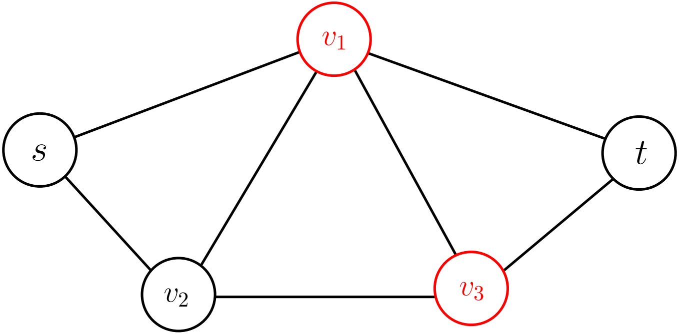


Menger's Theorem

Two fundamental duality results in (static) graph theory:

Theorem (Menger, 1927)

The maximum number of vertex-disjoint (edge-disjoint) s-t paths is equal to the minimum number of vertices (edges) needed to separate s from t.

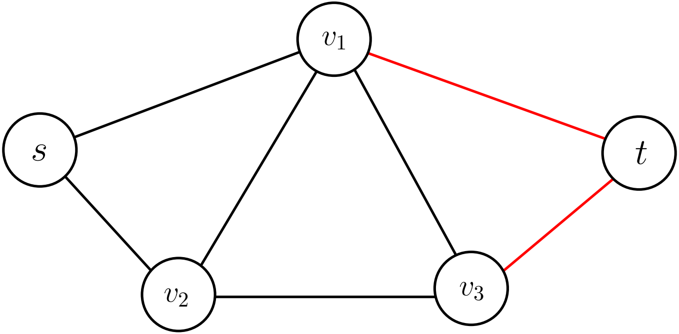


Menger's Theorem

Two fundamental duality results in (static) graph theory:

Theorem (Menger, 1927)

The maximum number of vertex-disjoint (edge-disjoint) s-t paths is equal to the minimum number of vertices (edges) needed to separate s from t.



Menger's Theorem

A temporal analogue of the "edge-version":

Theorem (Berman, 1996)

In single-labeled temporal graphs, the maximum number of edge-disjoint temporal s-t paths is equal to the minimum number of edges needed to temporally separate s from t.

Proof (sketch).

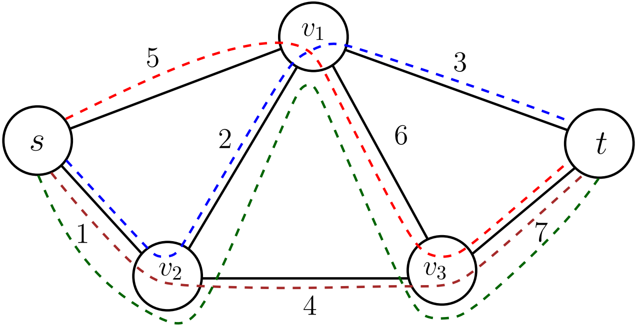
- Reduce the temporal (undirected) graph to a static directed graph.
- Edge-disjoint paths and minimum edge separators remain invariant.
- Apply Menger's edge-version theorem. □

Menger's Theorem

A temporal analogue of the "edge-version":

Theorem (Berman, 1996)

In single-labeled temporal graphs, the maximum number of edge-disjoint temporal s-t paths is equal to the minimum number of edges needed to temporally separate s from t.

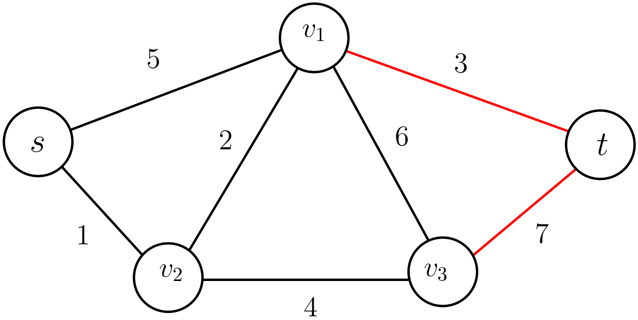


Menger's Theorem

A temporal analogue of the "edge-version":

Theorem (Berman, 1996)

In single-labeled temporal graphs, the maximum number of edge-disjoint temporal s-t paths is equal to the minimum number of edges needed to temporally separate s from t.



Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths $<$
minimum number of vertices needed to temporally separate s *from* t.

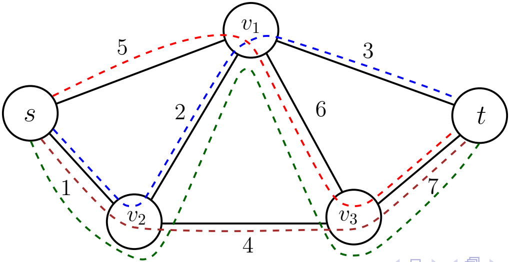
Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths $<$
minimum number of vertices needed to temporally separate s from t.

- no two vertex-disjoint temporal s-t paths



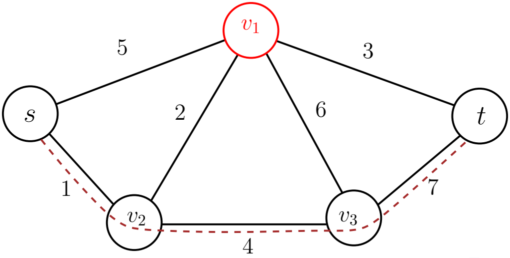
Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths <
minimum number of vertices needed to temporally separate s from t.

- no two vertex-disjoint temporal s-t paths
- the removal of any one vertex leaves s and t temporally connected



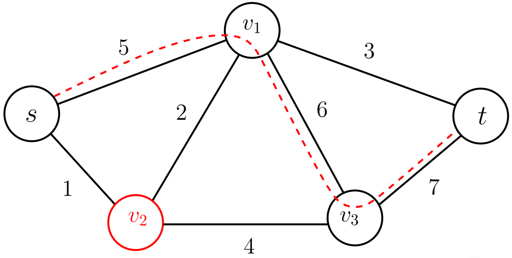
Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths <
minimum number of vertices needed to temporally separate s from t.

- no two vertex-disjoint temporal s-t paths
- the removal of any one vertex leaves s and t temporally connected



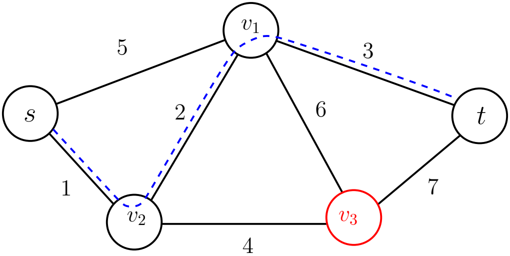
Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths $<$
minimum number of vertices needed to temporally separate s from t.

- no two vertex-disjoint temporal s-t paths
- the removal of any one vertex leaves s and t temporally connected



Menger's Theorem

However the "intuitive" temporal vertex-version fails:

Lemma (Berman, 1996; Kempe, Kleinberg, Kumar, 2000)

There exists a single-labeled temporal graph where:
maximum number of vertex-disjoint temporal s-t paths $<$
minimum number of vertices needed to temporally separate s *from* t.

Theorem (Kempe, Kleinberg, Kumar, 2000)

It is NP-hard to compute:

- *the maximum number of vertex-disjoint temporal s-t paths and*
- *the minimum number of vertices to temporally separate* s *from* t.

Menger's Theorem

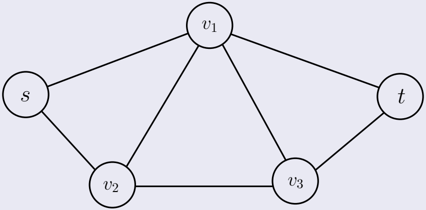
On the positive side:

- a single-labeled temporal graph is called Mengerian if the temporal vertex-version of Menger's theorem holds for every labeling λ

Similarly to Kuratowski's theorem:

Theorem (Kempe, Kleinberg, Kumar, 2000)

A single-labeled temporal graph (G, λ) is Mengerian $\Leftrightarrow$ G does not contain a subdivision of:



We say that:

- two temporal paths are out-disjoint if they never leave from the same node at the same time
- we remove departure time t from vertex u if:
  - we remove label t from for all edges (u, w)

We say that:

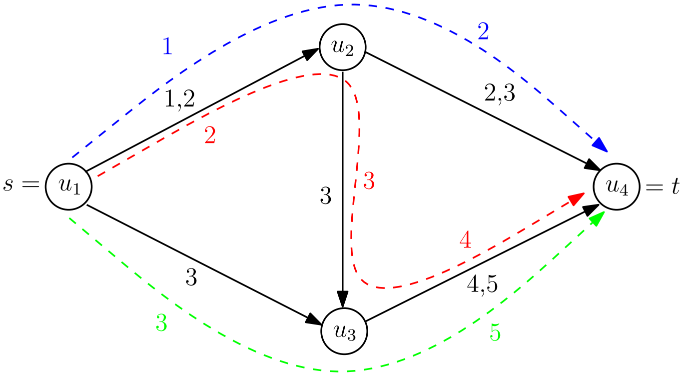
- two temporal paths are out-disjoint if they never leave from the same node at the same time
- we remove departure time t from vertex u if:
  - we remove label t from for all edges (u, w)

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, ICALP 2013)

In multi-labeled temporal graphs, the maximum number of out-disjoint s-t temporal paths equals the minimum number of vertex departure times needed to temporally separate s from t.

- vertex disjointness $\longrightarrow$ vertex departure time disjointness
- vertex removal $\longrightarrow$ vertex departure time removal

Three out-disjoint temporal paths from s to t:



Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, ICALP 2013)

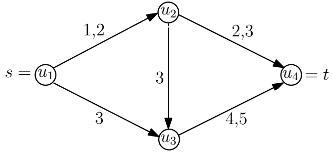
In multi-labeled temporal graphs, the maximum number of out-disjoint s-t temporal paths equals the minimum number of vertex departure times needed to temporally separate s from t.

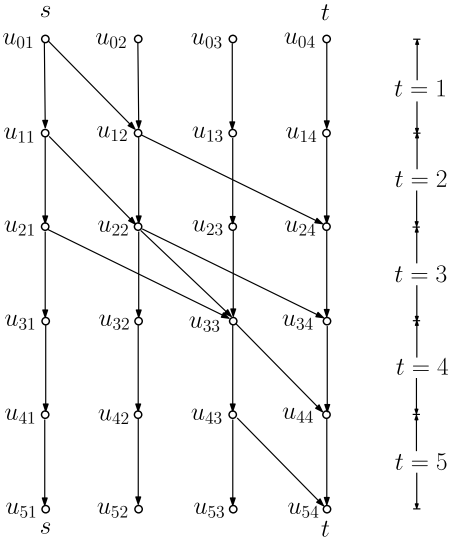
Proof (main idea).

- Using the time expansion of the temporal graph (G, λ)
  - static "layered" directed graph with one copy of G for every time label
- Equivalent (static) flow problem with capacities in $\{1, \lambda_{\max}\}$

Menger's Theorem

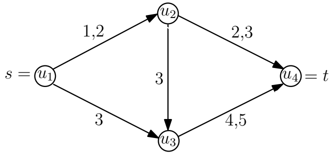
An appropriate temporal vertex-version of Menger's theorem

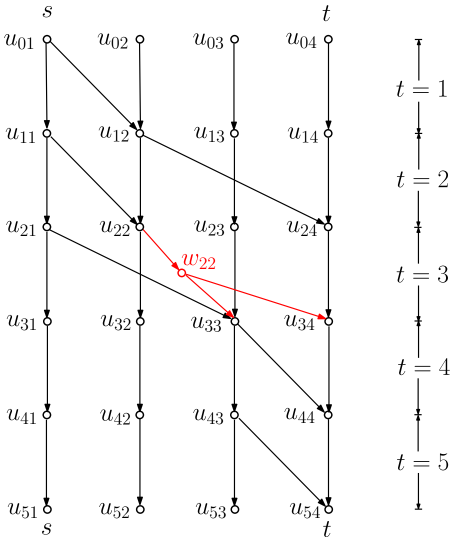




Menger's Theorem

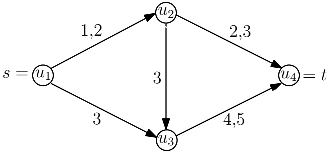
An appropriate temporal vertex-version of Menger's theorem

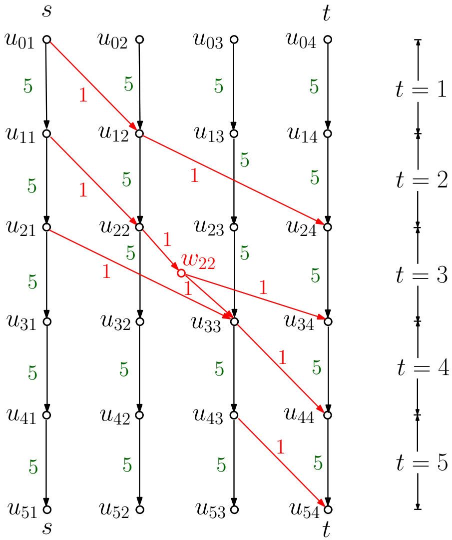




Menger's Theorem

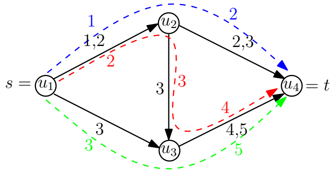
An appropriate temporal vertex-version of Menger's theorem

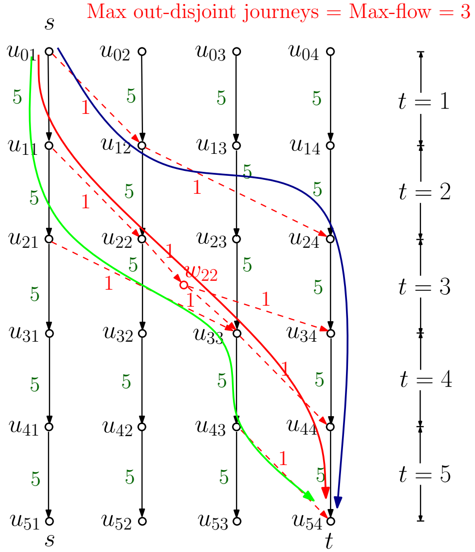




Menger's Theorem

An appropriate temporal vertex-version of Menger's theorem





Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Temporal design problems

So far:

- we were given the input temporal graph (G, λ) and
- we were asked to optimize some metric (e.g. a foremost path)

Many times the problem is different:

- we are given a graph G and
- we are asked to construct a time-labeling λ such that:
  - λ minimizes some cost function and
  - (G, λ) satisfies some connectivity constraints

[Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013]

[Akrida, Gasieniec, Mertzios, Spirakis, *WAOA*, 2015]

Temporal design problems

In many scheduling problems:

- the provided graph topology G represents a given static specification
  - e.g. available bus routes in the city center
- the aim is to organize a temporal schedule on this specification, e.g.
  - when the buses should be in which stop
  - such that every pair of stops is connected via a route
- while minimizing some cost function
  - e.g. with as few buses as possible

Temporal design problems

In many scheduling problems:

- the provided graph topology G represents a given static specification
  - e.g. available bus routes in the city center
- the aim is to organize a temporal schedule on this specification, e.g.
  - when the buses should be in which stop
  - such that every pair of stops is connected via a route
- while minimizing some cost function
  - e.g. with as few buses as possible

Creating and maintaining a connection does not come for free, e.g.:

- edge "rentals" / toll roads
- in wireless sensor networks the connection cost depends on the power consumption of the vertices awake

Temporal design problems

We mainly study the following cost functions of a time-label λ:

1. temporality τ: the maximum number of labels per edge
   - a distributed / decentralized measure of cost in the temporal network
2. temporal cost κ: the total number of labels on all edges
   - a centralized measure of cost
3. as well as trade-offs between the age $\alpha(\lambda)$ and these parameters

Temporal design problems

We mainly study the following cost functions of a time-label λ:

1. temporality τ: the maximum number of labels per edge
   - a distributed / decentralized measure of cost in the temporal network
2. temporal cost κ: the total number of labels on all edges
   - a centralized measure of cost
3. as well as trade-offs between the age $\alpha(\lambda)$ and these parameters

and two fundamental connectivity properties:

1. preserve in (G, λ) all reachabilities in G
   - if v is reachable from u in $G \Rightarrow u \rightsquigarrow v$ in (G, λ)
2. preserve in (G, λ) all paths in G
   - G has a path $P \Rightarrow (G, \lambda)$ has a temporal path on the same edges as P

Temporal design problems

We mainly study the following cost functions of a time-label λ:

1. temporality τ: the maximum number of labels per edge
   - a distributed / decentralized measure of cost in the temporal network
2. temporal cost κ: the total number of labels on all edges
   - a centralized measure of cost
3. as well as trade-offs between the age $\alpha(\lambda)$ and these parameters

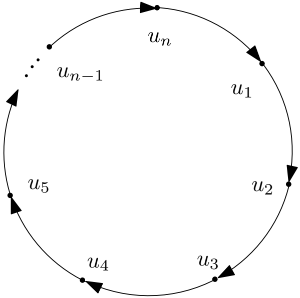
and two fundamental connectivity properties:

1. preserve in (G, λ) all reachabilities in G
   - if v is reachable from u in $G \Rightarrow u \rightsquigarrow v$ in (G, λ)
2. preserve in (G, λ) all paths in G
   - G has a path $P \Rightarrow (G, \lambda)$ has a temporal path on the same edges as P

Notation (combining cost function & connectivity property):

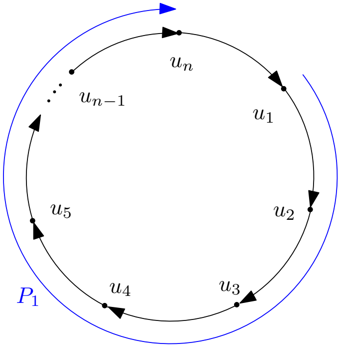
- $\tau(G, \textit{all paths})$, $\tau(G, \textit{all paths}, \alpha(\lambda))$, $\kappa(G, \textit{reach})$, etc.

Temporality of the ring C_n



Temporality of the ring C_n

- increasing labels on $P_1 \Rightarrow$ decreasing labels from (u_{n-1}, u_n) to (u_1, u_2)

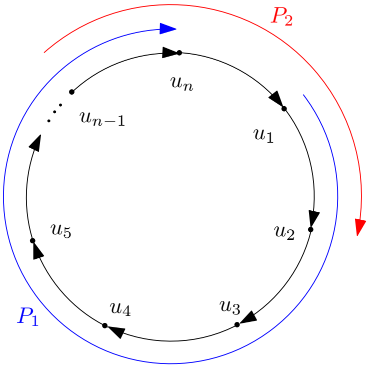


Temporality of the ring C_n

- increasing labels on P_1 $\Rightarrow$ decreasing labels from (u_{n-1}, u_n) to (u_1, u_2)
- P_2 uses first (u_{n-1}, u_n), then (u_1, u_2)

$\Rightarrow$ increasing pair of labels on these edges

- To preserve both P_1, P_2 we need 2 labels on at least one of these two edges $\Rightarrow$ $\tau(C_n, \textit{all paths}) \geq 2$

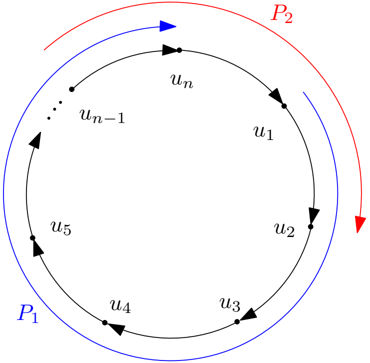


Temporality of the ring C_n

- increasing labels on P_1 $\Rightarrow$ decreasing labels from (u_{n-1}, u_n) to (u_1, u_2)
- P_2 uses first (u_{n-1}, u_n), then (u_1, u_2)

$\Rightarrow$ increasing pair of labels on these edges

- To preserve both P_1, P_2 we need 2 labels on at least one of these two edges $\Rightarrow$ $\tau(C_n, all\ paths) \geq 2$



- The labeling that assigns to each edge (u_i, u_{i+1}) the labels $\{i, n+i\}$ preserves all simple paths, i.e. $\tau(C_n, all\ paths) \leq 2$

$\Rightarrow$ $\tau(C_n, all\ paths) = 2$

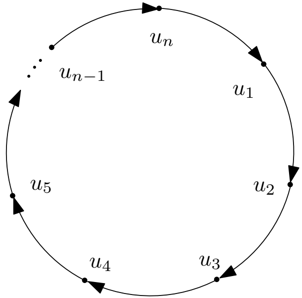
- The maximum label is $2n$ (can be "tuned" to $2n-2$)

Temporality of the ring C_n

Restricting the age

What if we restrict the age to $\alpha(\lambda) = n - 1$?

- Assume that some edge e of C_n misses label $i \in \{1, 2, \ldots, n-1\}$
- Then there exists a temporal path on C_n that needs label i on edge e to finish by time $n - 1$

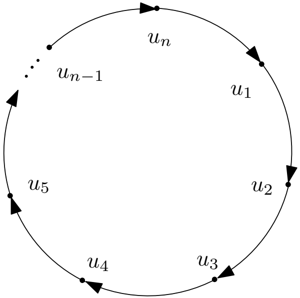


What if we restrict the age to $\alpha(\lambda) = n - 1$?

- Assume that some edge e of C_n misses label $i \in \{1, 2, \ldots, n-1\}$
- Then there exists a temporal path on C_n that needs label i on edge e to finish by time $n - 1$

$\Rightarrow$ the optimal labeling assigns $\{1, 2, , \ldots, n-1\}$ to all edges of C_n

$\Rightarrow$ $\tau(C_n, all\ paths, n-1) = n - 1$



More generally:

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

If G is a directed ring C_n and $\alpha(\lambda) = (n-1) + k$, where $1 \leq k \leq n-1$, then

$$\tau(G, \textit{all paths}, \alpha) = \Theta(n/k).$$

In particular: $\lfloor \frac{n-1}{k+1} \rfloor + 1 \leq \tau(G, \textit{all paths}, \alpha) \leq \lceil \frac{n}{k+1} \rceil + 1.$

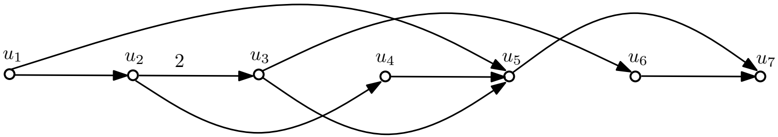
Temporality of a DAG

A topological sort of a digraph G:

- a linear ordering of its vertices, where
- if G contains an arc (u, v) then u appears before v

It is known:

- a digraph G can be topologically sorted $\Leftrightarrow$ G is a DAG



Temporality of a DAG

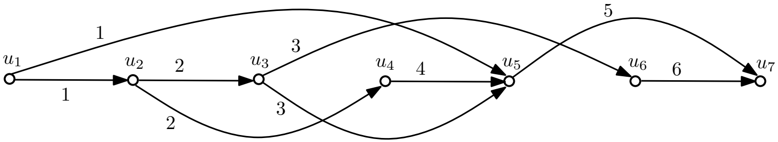
Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

If G is a DAG then $\tau(G, \text{all paths}) = 1$.

Proof.

- Take a topological sort $u_1, u_2, \ldots, u_n$ of G
- Give to every label i to every edge (u_i, u_j), where $i < j$.

$\square$



Temporality of a DAG

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

If G is a DAG then $\tau(G, \textit{all paths}) = 1$.

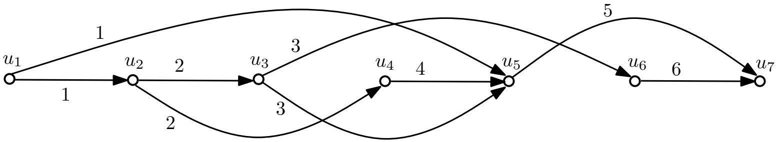
Proof.

- Take a topological sort $u_1, u_2, \ldots, u_n$ of G
- Give to every label i to every edge (u_i, u_j), where $i < j$.

$\square$

Remarks:

- $\tau(G, \textit{all paths}) = 1$
- the same even if we restrict the age



A generic method for lower bounds of temporality

Intuition gained:

- cycles can increase the temporality of a (di)graph

A generic method for lower bounds of temporality

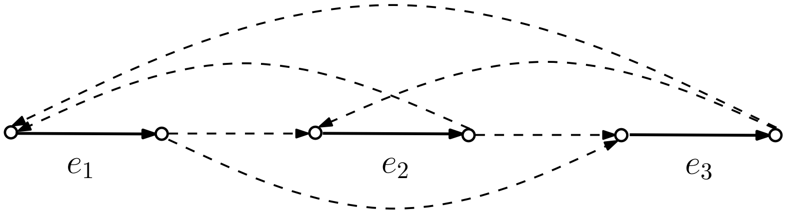
Intuition gained:

- cycles can increase the temporality of a (di)graph

Based on this intuition:

Definition (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

The set $K = \{e_1, e_2, \ldots, e_k\} \subseteq E(G)$ is an edge-kernel of G if for every permutation of K there is a (static) path of G that visits all edges of K according to this permutation.



A generic method for lower bounds of temporality

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

If a (di)graph G has an edge-kernel of size k then $\tau(G, \text{all paths}) \geq k$.

A generic method for lower bounds of temporality

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

If a (di)graph G has an edge-kernel of size k then $\tau(G, \text{all paths}) \geq k$.

Proof.

- Let K be a kernel of size k.
- On every edge $e \in K$, sort the labels increasingly.
- Construct a permutation π of K as follows:
  - e_1 is an edge with the maximum 1st label in K,
  - e_2 is an edge with the maximum 2nd label in K, ...
  - e_k is an edge with the maximum kth label in K.
- In any temporal path that visits the edges according to π:
  - at the ith edge e_i of K we can not use any of its $i-1$ smallest labels

  $\Rightarrow$ edge e_i needs one ith label

$\Rightarrow$ at least one edge of K needs at least kth labels $\square$

A generic method for lower bounds of temporality

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

The complete (di)graph on n vertices has an edge-kernel of size $\lfloor n/2 \rfloor$.

A generic method for lower bounds of temporality

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

The complete (di)graph on n vertices has an edge-kernel of size $\lfloor n/2 \rfloor$.

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

There exist (undirected) planar graphs with an edge-kernel of size $\Omega(n^{\frac{1}{3}})$.

A generic method for lower bounds of temporality

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

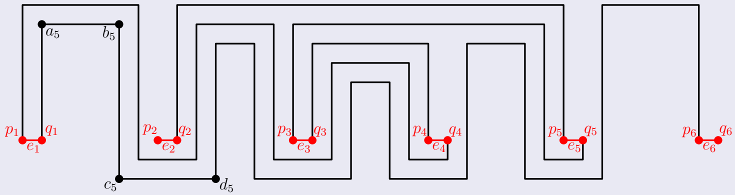
The complete (di)graph on n vertices has an edge-kernel of size $\lfloor n/2 \rfloor$.

Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

There exist (undirected) planar graphs with an edge-kernel of size $\Omega(n^{\frac{1}{3}})$.

Proof idea.

- Consider the grid graph $G = G_{2n^2,2n}$ with $O(n^3)$ vertices and edges
- Then G has an edge-kernel of n edges:



□

Temporality: preserving all reachabilities

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let G be an undirected (or strongly connected directed) graph.
Then $\tau(G, reach) \leq 2$.

Proof.

- pick an arbitrary vertex v
- let v have (static) distance at most k to all other vertices
- build a temporal in-tree to vertex v with labels $\{1, 2, \ldots, k\}$
- from v build a temporal out-tree to vertex v with labels $\{k+1, k+2, \ldots, 2k\}$

$\Rightarrow$ all vertices remain temporally connected with 2 labels per edge

$\square$

Temporality: preserving all reachabilities

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let G be an undirected (or strongly connected directed) graph. Then $\tau(G, reach) \leq 2$.

Similarly to our analysis for DAGs:

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let G be a directed graph. Then $\tau(G, reach) = \max_{C \in \mathcal{C}(G)} \tau(C, reach)$, where $\mathcal{C}(G)$ is the set of strongly connected components of G.

Temporality: preserving all reachabilities

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let G be an undirected (or strongly connected directed) graph. Then $\tau(G, reach) \leq 2$.

Similarly to our analysis for DAGs:

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let G be a directed graph. Then $\tau(G, reach) = \max_{C \in \mathcal{C}(G)} \tau(C, reach)$, where $\mathcal{C}(G)$ is the set of strongly connected components of G.

Therefore:

Corollary

$\tau(G, reach) \leq 2$ for every directed or undirected graph.

That is: we can preserve all reachabilities with at most 2 labels per edge.

Temporal cost for preserving all reachabilities

A very different cost function: total number κ of labels

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let $d(G)$ denote the (static) diameter of the directed graph G. The problem of computing $\kappa(G, reach, d(G))$ is APX-hard, even when each directed cycle of G has length at most 2.

Temporal cost for preserving all reachabilities

A very different cost function: total number κ of labels

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

Let $d(G)$ denote the (static) diameter of the directed graph G. The problem of computing $\kappa(G, reach, d(G))$ is APX-hard, even when each directed cycle of G has length at most 2.

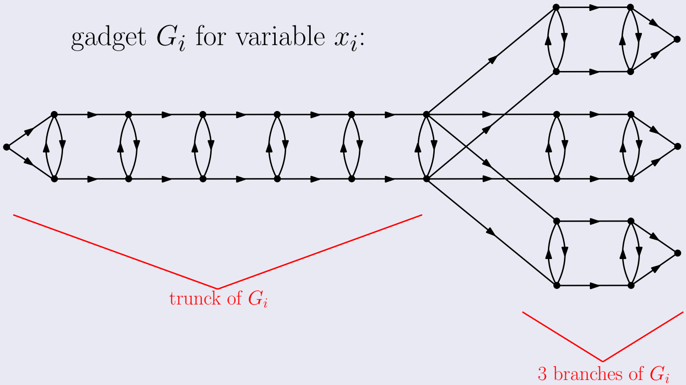
Proof (sketch).

- reduction from Max-XOR(3):
  - formula ϕ with n variables and m clauses
  - XOR-clauses $(\ell_i \oplus \ell_j)$ with two literals each: $(\ell_i \oplus \ell_j) = 1 \;\Leftrightarrow\; \ell_i \neq \ell_j$
  - each variable appears in at most 3 clauses $\;\Rightarrow\;$ $m \leq \frac{3}{2}n$
  - the goal is to find a truth assignment τ with the maximum number $|\tau(\phi)|$ of XOR-satisfied clauses
- from ϕ we construct a graph G_ϕ and we prove:
  - $|\tau(\phi)| \geq k \;\Leftrightarrow\; \kappa(G_\phi, reach, d(G_\phi)) \leq 39n - 4m - 2k$

Temporal cost for preserving all reachabilities

Proof (sketch, continued).

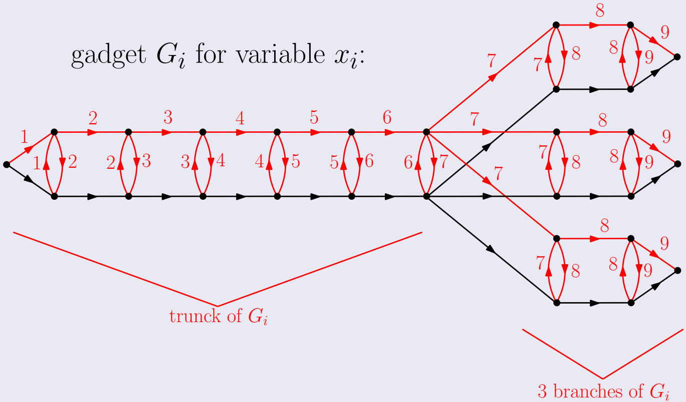
- diameter $d(G_i) = 9$



Temporal cost for preserving all reachabilities

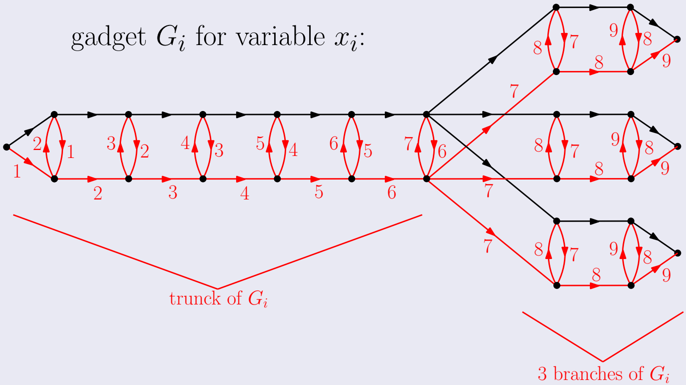
Proof (sketch, continued).

- diameter $d(G_i) = 9$
- to achieve a maximum label 9 we have two choices:
  - $x_i = 0$



Proof (sketch, continued).

- diameter $d(G_i) = 9$
- to achieve a maximum label 9 we have two choices:
  - $x_i = 0$
  - $x_i = 1$

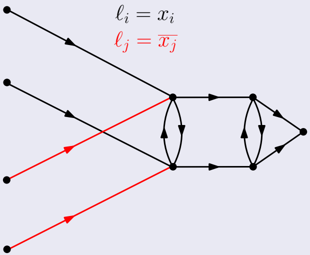


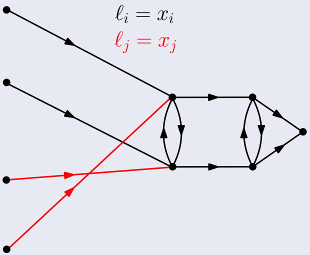
Temporal cost for preserving all reachabilities

Proof (sketch, continued).

- for every clause $(\ell_i \oplus \ell_j)$ where:
  - ℓ_i corresponds to the pth appearance of x_i $(p \in \{1, 2, 3\})$
  - ℓ_j corresponds to the qth appearance of x_j $(q \in \{1, 2, 3\})$

  we identify the pth branch of G_i and the qth branch of G_j as follows:





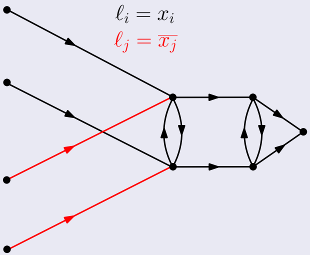
□

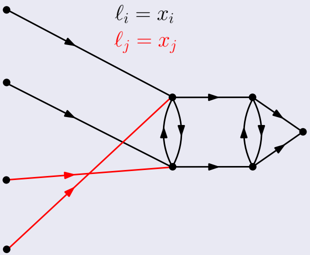
Temporal cost for preserving all reachabilities

Proof (sketch, continued).

- for every clause $(\ell_i \oplus \ell_j)$ where:
  - ℓ_i corresponds to the pth appearance of x_i $\quad (p \in \{1, 2, 3\})$
  - ℓ_j corresponds to the qth appearance of x_j $\quad (q \in \{1, 2, 3\})$

  we identify the pth branch of G_i and the qth branch of G_j as follows:





- $\ell_i \neq \ell_j \;\Leftrightarrow\;$ the correct "tracks" of these branches are labeled
- otherwise we use both "tracks" $\;\Rightarrow\;$ pay more labels $\square$

Temporal cost for preserving all reachabilities

A simple approximation algorithm:

- the reachability number of $u \in V$:

$$r(u) = |\{v \in V : v \text{ is reachable from } u\}|$$

- the total reachability number: $r(G) = \sum_{u \in V} r(u)$

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

A $\frac{r(G)}{n-1}$-approximation for $\kappa(G, reach, d(G))$ can be computed in polynomial time for connected graphs G.

Temporal cost for preserving all reachabilities

A simple approximation algorithm:

- the reachability number of $u \in V$:

$$r(u) = |\{v \in V : v \text{ is reachable from } u\}|$$

- the total reachability number: $r(G) = \sum_{u \in V} r(u)$

Theorem (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

A $\frac{r(G)}{n-1}$-approximation for $\kappa(G, reach, d(G))$ can be computed in polynomial time for connected graphs G.

Proof.

- Compute from every $u \in V$ a temporal out-tree

$\Rightarrow$ all reachabilities are maintained with $\leq r(G)$ labels

- OPT$\geq n-1$ $\Rightarrow$ approximation ratio $\frac{r(G)}{n-1}$ $\square$

Removal cost for preserving all reachabilities

The "inverse" design problem:

- given a temporal graph (G, λ) that maintains all reachabilities of G
- remove the maximum number of labels by maintaining reachabilities
- removal cost $r(G, \lambda)$

Removal cost for preserving all reachabilities

The "inverse" design problem:

- given a temporal graph (G, λ) that maintains all reachabilities of G
- remove the maximum number of labels by maintaining reachabilities
- removal cost $r(G, \lambda)$

Theorem (Akrida, Gasieniec, Mertzios, Spirakis, *WAOA*, 2015)

The problem of computing $r(G, \lambda)$ is APX-hard on undirected graphs G.

Removal cost for preserving all reachabilities

The "inverse" design problem:

- given a temporal graph (G, λ) that maintains all reachabilities of G
- remove the maximum number of labels by maintaining reachabilities
- removal cost $r(G, \lambda)$

Theorem (Akrida, Gasieniec, Mertzios, Spirakis, *WAOA*, 2015)

The problem of computing $r(G, \lambda)$ is APX-hard on undirected graphs G.

Proof (sketch).

- reduction from monotone Max-XOR(3):
  - same as Max-XOR(3) but no variable is negated
- from ϕ we construct a graph G_ϕ and we prove:
  - $|\tau(\phi)| \geq k \;\Leftrightarrow\; r(G_\phi, \lambda) \geq 9n + k$

$\square$

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Temporal exploration

Temporal Exploration Problem (TEXP) (Michail, Spirakis, 2014)

Input: *Temporal graph* (G, λ) *and source vertex* s

Goal: *Visit each vertex at least once with a temporal walk that minimizes the arrival time (possibly revisiting vertices)*

Its "static analogue": Graphic Traveling Salesman Problem

- $\frac{13}{9}$-approximation algorithm [Mucha, *Th. Comp. Syst.*, 2014]

Temporal exploration

Temporal Exploration Problem (TEXP) (Michail, Spirakis, 2014)

Input: *Temporal graph* (G, λ) *and source vertex* s

Goal: *Visit each vertex at least once with a temporal walk that minimizes the arrival time (possibly revisiting vertices)*

Its "static analogue": Graphic Traveling Salesman Problem

- $\frac{13}{9}$-approximation algorithm [Mucha, *Th. Comp. Syst.*, 2014]

Observation

The decision version in the static case can be solved in linear time.

Proof.

- A static graph G is explorable $\Leftrightarrow$ G is connected.

$\Rightarrow$ Check connectivity in G by DFS.

$\square$

Temporal exploration

Temporal Exploration Problem (TEXP) (Michail, Spirakis, 2014)

Input: *Temporal graph* (G, λ) *and source vertex* s

Goal: *Visit each vertex at least once with a temporal walk that minimizes the arrival time (possibly revisiting vertices)*

Its "static analogue": Graphic Traveling Salesman Problem

- $\frac{13}{9}$-approximation algorithm [Mucha, *Th. Comp. Syst.*, 2014]

Observation

If a temporal graph (G, λ) *is connected at every time* t*, then it is always explorable.*

Temporal exploration

However:

Theorem (Michail, Spirakis, *MFCS*, 2014)

The decision version in the temporal case is NP-complete.

Proof.

Clearly in NP:

- certificate is a temporal walk visiting all vertices
- concatenation of $n-1$ temporal paths
- in worst case: each temporal path re-visits all visited vertices

$\Rightarrow$ size of certificate: $O(n^2)$ (regardless of the arrival time)

Proof (continued).

NP-hardness: reduction from HAMILTONIAN PATH (HP)

- given a graph G and source s (instance of (HP))
- construct a temporal graph D on the same vertices and edges as G:
  - where $\lambda(e) = \{1, 2, \ldots, n-1\}$ for every edge of G

Proof (continued).

NP-hardness: reduction from HAMILTONIAN PATH (HP)

- given a graph G and source s (instance of (HP))
- construct a temporal graph D on the same vertices and edges as G:
  - where $\lambda(e) = \{1, 2, \ldots, n-1\}$ for every edge of G

1. Let G have a Hamiltonian path starting from s

$\Rightarrow$ D has a temporal Hamiltonian path from s, visiting all vertices until time $n-1$

$\Rightarrow$ D is explorable from s

Temporal exploration

Proof (continued).

NP-hardness: reduction from Hamiltonian Path (HP)

- given a graph G and source s (instance of (HP))
- construct a temporal graph D on the same vertices and edges as G:
  - where $\lambda(e) = \{1, 2, \ldots, n-1\}$ for every edge of G

2. Conversely, let D have a temporal walk W from s, visiting each vertex at least once

- Since the age of D is $n-1$:
  - W must visit all vertices in $n-1$ steps
  - $\Rightarrow$ it visits each vertex exactly once

$\Rightarrow$ G has a Hamiltonian path from s

$\square$

Temporal exploration

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

There exists an infinite family of temporal graphs that require $\Omega(n^2)$ time to be explored.

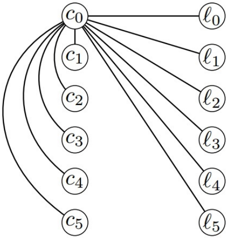
Temporal exploration

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

There exists an infinite family of temporal graphs that require $\Omega(n^2)$ time to be explored.

Proof.

- Let $V = \{c_j, \ell_j : 0 \leq j \leq n-1\}$ be the vertex set of G
- The "snapshot" of G at time $t \geq 0$ is a star with center $c_{t \bmod n}$



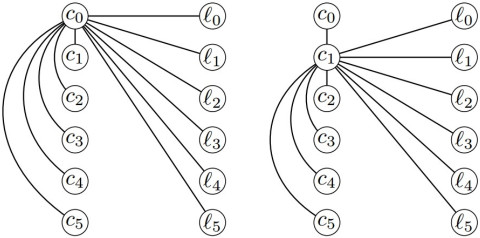
Temporal exploration

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

There exists an infinite family of temporal graphs that require $\Omega(n^2)$ time to be explored.

Proof.

- Let $V = \{c_j, \ell_j : 0 \leq j \leq n-1\}$ be the vertex set of G
- The "snapshot" of G at time $t \geq 0$ is a star with center $c_{t \bmod n}$

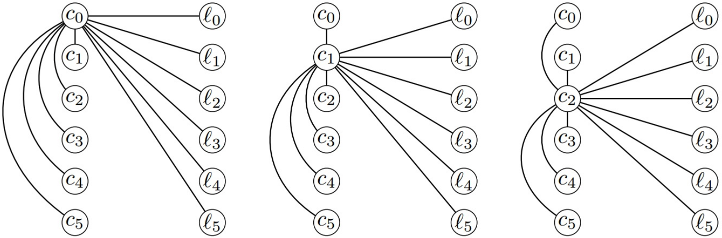


Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

There exists an infinite family of temporal graphs that require $\Omega(n^2)$ time to be explored.

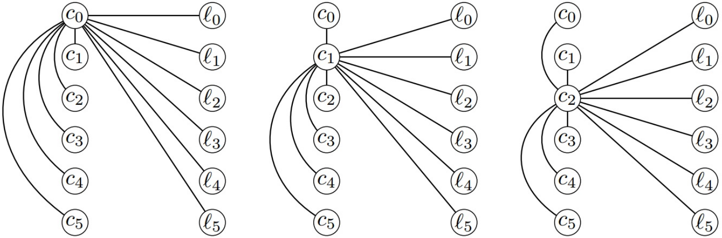
Proof.

- Let $V = \{c_j, \ell_j : 0 \leq j \leq n-1\}$ be the vertex set of G
- The "snapshot" of G at time $t \geq 0$ is a star with center $c_{t \bmod n}$



Temporal exploration

Proof (continued).

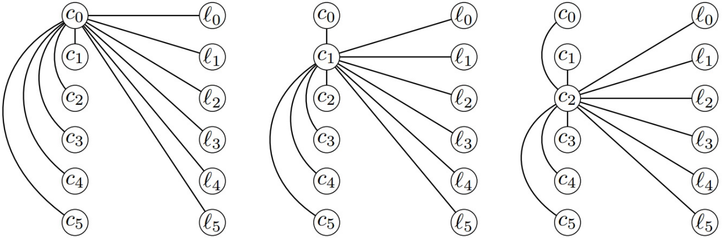


- If the exploring agent is at a vertex that is not the current center:
  - it can only wait or travel to the current center
- If it moves, at the next step it will be again not in the current center

$\Rightarrow$ to go from ℓ_i to ℓ_j, $i \neq j$, n steps are needed:
  - the fastest way is to move from ℓ_i to the current center, to wait $n-1$ steps, and then go to ℓ_j

Temporal exploration

Proof (continued).



- If the exploring agent is at a vertex that is not the current center:
  - it can only wait or travel to the current center
- If it moves, at the next step it will be again not in the current center

$\Rightarrow$ to go from ℓ_i to ℓ_j, $i \neq j$, n steps are needed:

  - the fastest way is to move from ℓ_i to the current center, to wait $n-1$ steps, and then go to ℓ_j

$\Rightarrow$ the total number of steps is $\Omega(n^2)$ □

Temporal exploration

Modifying the previous reduction, the result can be strongly amplified:

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

Approximating TEXP with ratio $O(n^{1-\varepsilon})$ is NP-hard.

Temporal exploration

Modifying the previous reduction, the result can be strongly amplified:

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

Approximating TEXP with ratio $O(n^{1-\varepsilon})$ is NP-hard.

The previous reduction can be also "parameterized":

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

For every $\Delta > 0$, there exists an infinite family of temporal graphs with maximum degree Δ that require $\Omega(\Delta n)$ time to be explored.

Temporal exploration

Modifying the previous reduction, the result can be strongly amplified:

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

Approximating TEXP with ratio $O(n^{1-\varepsilon})$ is NP-hard.

The previous reduction can be also "parameterized":

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

For every $\Delta > 0$, there exists an infinite family of temporal graphs with maximum degree Δ that require $\Omega(\Delta n)$ time to be explored.

Furthermore, on restricted classes of underlying graphs:

Theorem (Erlebach, Hoffmann, Kammer, *ICALP*, 2015)

Any temporal graph whose underlying graph has treewidth at most k, can be explored in $O(n^{1.5} k^2 \log n)$ time.

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Temporal Traveling Salesman Problem

A different kind of graph dynamicity:

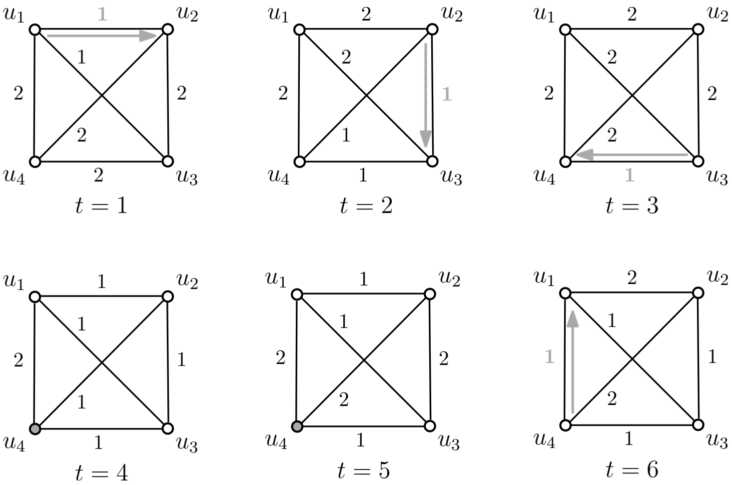
Temporal TSP(1,2) (Michail, Spirakis, 2014)

Input: *Temporal graph (G, λ) which is always complete and edge weights among $\{1, 2\}$ which vary with time*

Goal: *Find a temporal tour of (G, λ) with the minimum cost*

Temporal Traveling Salesman Problem

Example:



Temporal Traveling Salesman Problem

The solution approach to approximate TTSP(1,2):
[Michail, Spirakis, *MFCS*, 2014]

- compute a "temporal matching" M with as many 1's as possible
- patch M with the remaining edges in a time-respecting way

where a temporal matching of (G, λ) is:

- a collection $M = \{(e_1, \ell_1), \ldots, (e_k, \ell_k)\}$, where
- $\{e_1, \ldots, e_k\}$ is a matching in G and $\ell_i \in \lambda(e_i)$ for every i,
- $\ell_i \neq \ell_j$ for $i \neq j$.

Temporal Traveling Salesman Problem

The solution approach to approximate TTSP(1,2):
[Michail, Spirakis, *MFCS*, 2014]

- compute a "temporal matching" M with as many 1's as possible
- patch M with the remaining edges in a time-respecting way

where a temporal matching of (G, λ) is:

- a collection $M = \{(e_1, \ell_1), \ldots, (e_k, \ell_k)\}$, where
- $\{e_1, \ldots, e_k\}$ is a matching in G and $\ell_i \in \lambda(e_i)$ for every i,
- $\ell_i \neq \ell_j$ for $i \neq j$.

The Max-TEM(k) problem is, given a temporal graph:

- to compute a temporal matching of maximum size, where
- $|\ell_i - \ell_j| \geq k$ whenever $i \neq j$.

Temporal Traveling Salesman Problem

Theorem (Michail, Spirakis, 2014)

Max-TEM(k) is NP-complete for every $k \geq 1$.

Proof: by a reduction from BALANCED 3SAT □

Temporal Traveling Salesman Problem

Theorem (Michail, Spirakis, 2014)

Max-TEM(k) is NP-complete for every $k \geq 1$.

Proof: by a reduction from BALANCED 3SAT $\square$

On the positive side:

Lemma (Michail, Spirakis, 2014)

An $\frac{1}{c}$*-approximation for Max-TEM(2) implies a* $(2 - \frac{1}{2c})$*-approximation for TTSP(1,2).*

Temporal Traveling Salesman Problem

Lemma (Michail, Spirakis, 2014)

There is a $\frac{1}{2+\varepsilon}$-approximation algorithm for Max-TEM(2).

Proof.

- From the time expansion of (G, λ) build a "conflict graph" G^* which is a 5-claw free graph
- Max-TEM(2) in (G, λ) $\Leftrightarrow$ independent sets in G^*
- Apply the algorithm of [Halldórsson, *SODA*, 1995]

$\square$

Temporal Traveling Salesman Problem

Lemma (Michail, Spirakis, 2014)

There is a $\frac{1}{2+\varepsilon}$-approximation algorithm for Max-TEM(2).

Proof.

- From the time expansion of (G, λ) build a "conflict graph" G^* which is a 5-claw free graph
- Max-TEM(2) in (G, λ) $\Leftrightarrow$ independent sets in G^*
- Apply the algorithm of [Halldórsson, *SODA*, 1995]

$\square$

Therefore:

Corollary (Michail, Spirakis, 2014)

There is a $(\frac{7}{4} + \varepsilon)$-approximation algorithm for TTSP(1,2).

Overview

- Temporal graphs
- Temporal paths
- Strongly connected components
- Menger's theorem
- Temporal design problems
- Temporal exploration
- Temporal TSP
- Future research directions

Research Directions

- Find constant-factor approximations for the various temporal graph design problems
- Other structural properties of G that cause a growth of temporality for "all paths"? (apart from edge-kernels)
  - algorithms / complexity?
- Other natural connectivity properties subject to which optimization is to be performed
- Efficient deterministic/randomized/approximation algorithms on special temporal graph classes, i.e. by restricting:
  - the underlying topology G and/or
  - the temporal pattern with which the time-labels appear (a new dimension with no previous static analogue!)

Research Directions

- Temporal graphs defined by the mobility patterns of mobile wireless entities modeled by a sequence of unit disk graphs
  - Well-motivated as a natural source of temporal graphs
  - May allow for better approximations
- Other natural non-path temporal problems (apart from matchings)
  - a recently defined notion of a "Δ-temporal clique" in social networks: *"a set of nodes and a time interval such that all pairs interact at least every Δ during this interval"* [Viard, Latapy, Magnien, *ASONAM*, 2015]
- Our results so far are a first step towards answering this fundamental question:

To what extent can algorithmic and structural results of graph theory be carried over to temporal graphs?

Thank you for your attention!