Algorithms and Complexity on Temporal Graphs

George B. Mertzios

Department of Computer Science,
Durham University, UK

GROW 2017

Fields Institute
University of Toronto

October 2017

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

$\Rightarrow$ 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

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

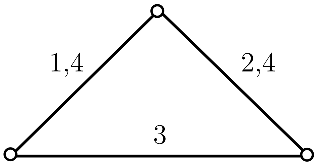
Formally:

Definition (Temporal Graph)

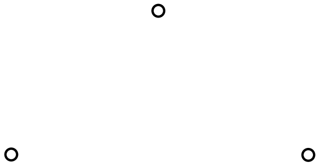
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



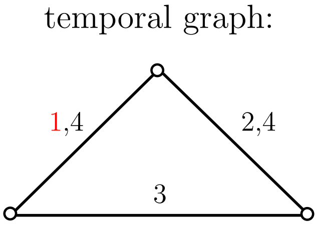
Temporal graphs

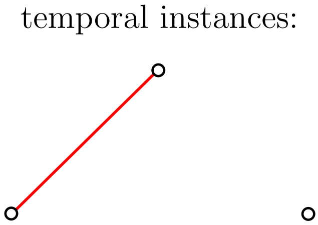
Formally:

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 graphs

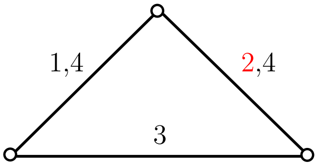
Formally:

Definition (Temporal Graph)

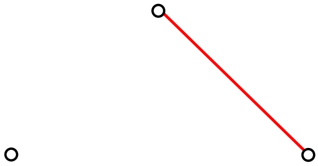
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

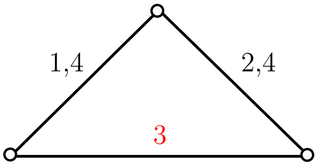
Formally:

Definition (Temporal Graph)

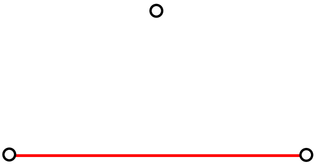
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

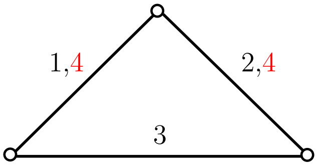
Formally:

Definition (Temporal Graph)

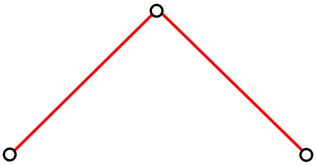
A temporal graph is a pair (G, λ) where:

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

temporal graph:



temporal instances:



Temporal graphs

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
- Temporal Networks [Holme, Saramäki, eds., *Springer*, 2013]
  - temporal network methods for complex networks
- 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

Overview

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

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

Intuition:

- information "flows" along edges whose labels respect time ordering

Most known 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:

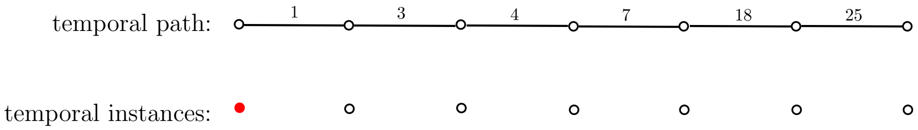
Definition (Temporal path; Time-respecting path; Journey)

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:



The most natural known temporal notion in temporal graphs:

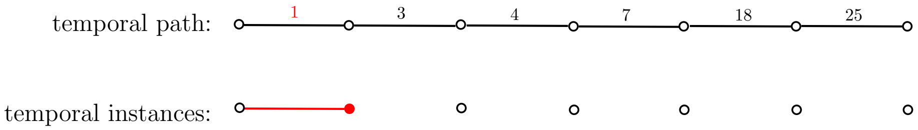
Definition (Temporal path; Time-respecting path; Journey)

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:

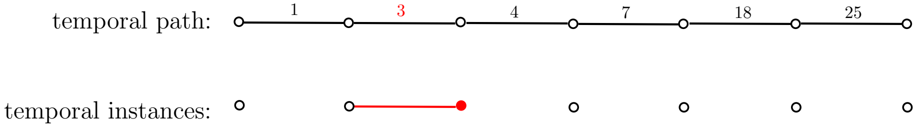
Definition (Temporal path; Time-respecting path; Journey)

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:

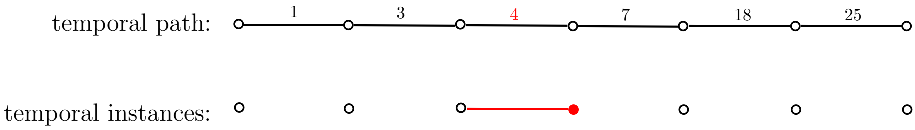
Definition (Temporal path; Time-respecting path; Journey)

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:

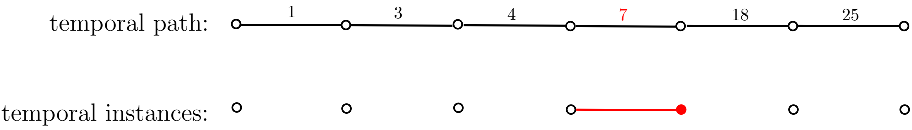
Definition (Temporal path; Time-respecting path; Journey)

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:

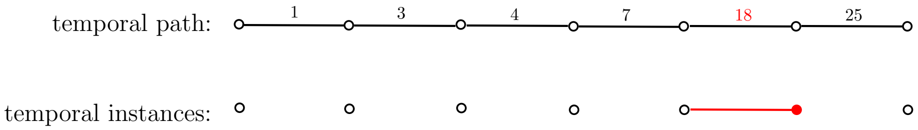
Definition (Temporal path; Time-respecting path; Journey)

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:

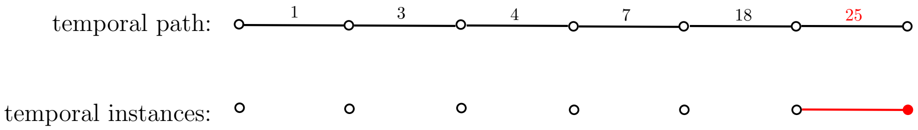
Definition (Temporal path; Time-respecting path; Journey)

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:

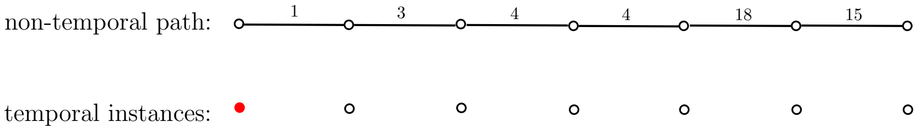
Definition (Temporal path; Time-respecting path; Journey)

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:



Temporal paths

The most natural known temporal notion in temporal graphs:

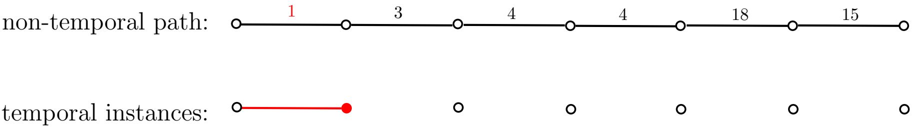
Definition (Temporal path; Time-respecting path; Journey)

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:

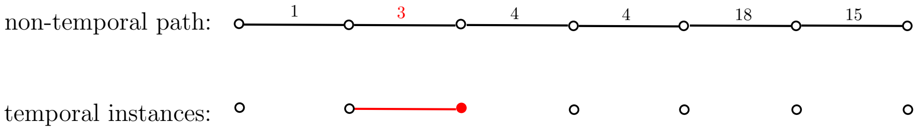
Definition (Temporal path; Time-respecting path; Journey)

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:

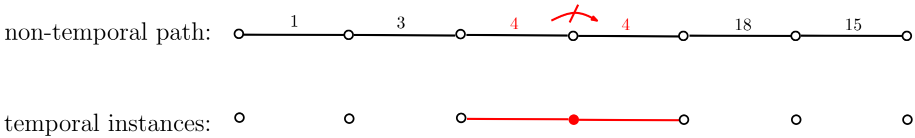
Definition (Temporal path; Time-respecting path; Journey)

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:

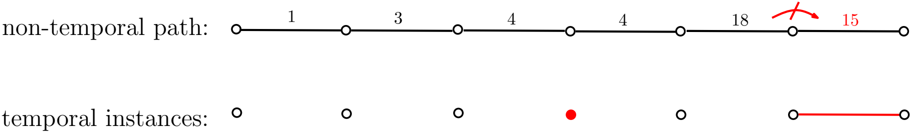
Definition (Temporal path; Time-respecting path; Journey)

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:

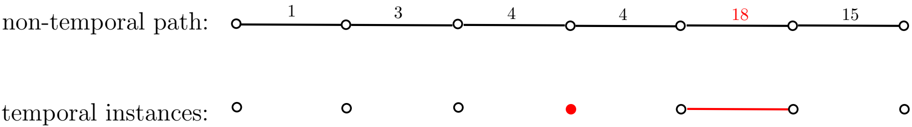
Definition (Temporal path; Time-respecting path; Journey)

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?

Answer: Not uniquely defined!

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

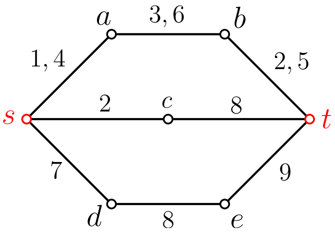
Metrics to optimize

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:



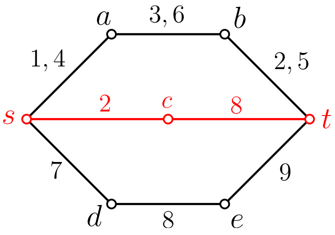
Metrics to optimize

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)

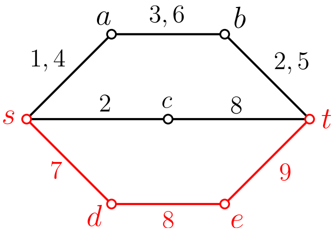
Metrics to optimize

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)

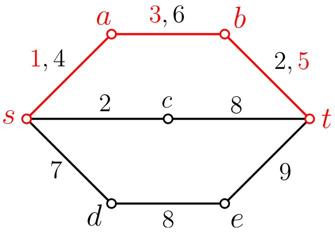
Metrics to optimize

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 5)

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

Metrics to optimize

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
$R \leftarrow \{s\}$
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

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 for computing:

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

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

However: Not all "path-related" temporal problems tractable, e.g. some temporal variations of:

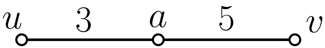
- connectivity problems
- reachability problems

Overview

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

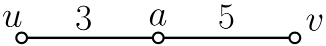
Temporal strongly connected components

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

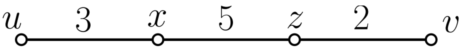


Temporal strongly connected components

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



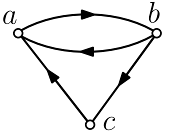
- and not transitive: $u \rightsquigarrow z,\; z \rightsquigarrow v \;\not\Rightarrow\; u \rightsquigarrow v$



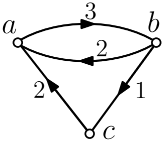
$\Rightarrow$ the time dimension creates its own "level of direction"

Temporal strongly connected components

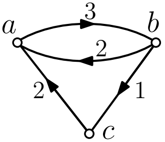
static:



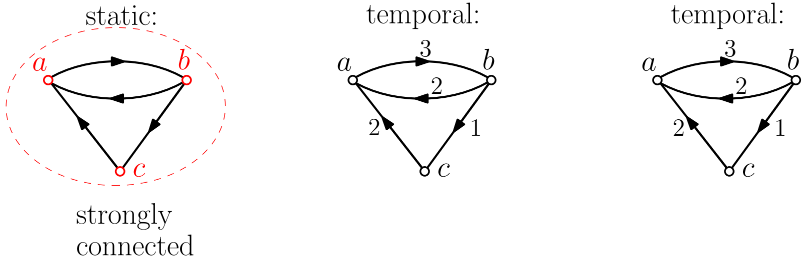
temporal:



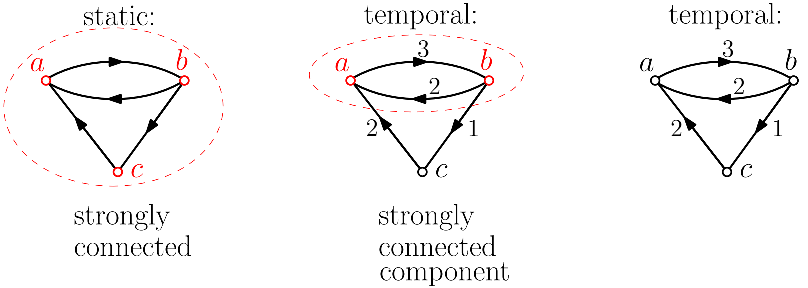
temporal:



Temporal strongly connected components

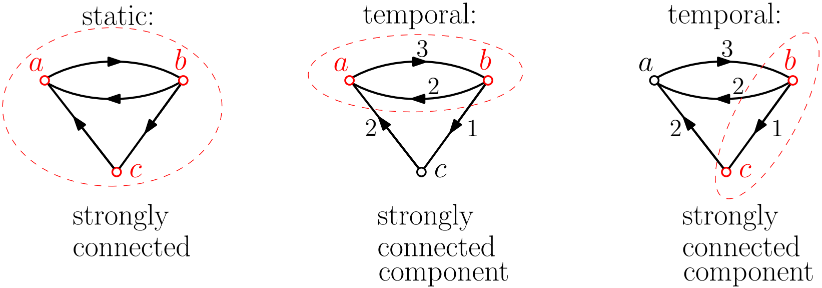


Temporal strongly connected components



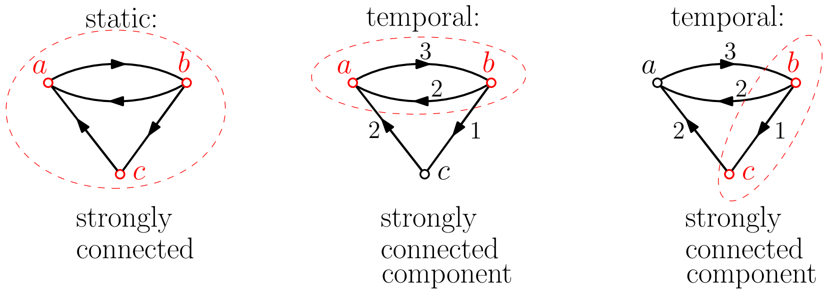
- $\{a, b\}$: direct temporal paths between a and b

Temporal strongly connected components



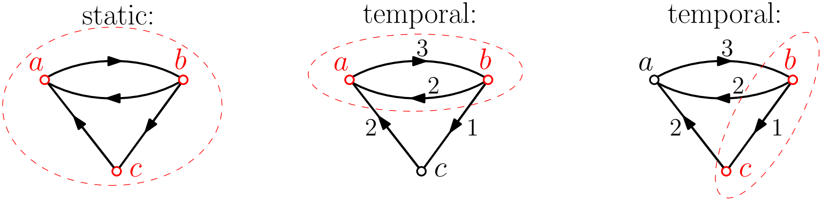
- $\{a, b\}$: direct temporal paths between a and b
- $\{b, c\}$: the only temporal path from c to b passes through $a \notin \{b, c\}$

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

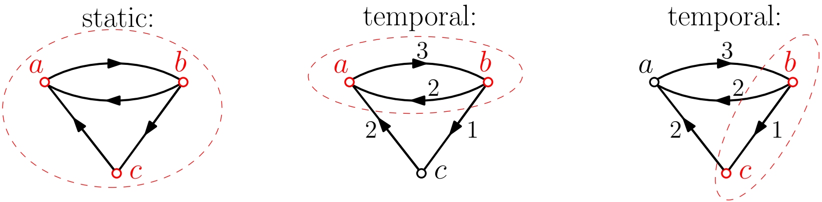


Definition (Bharda, Ferreira, 2003)

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)

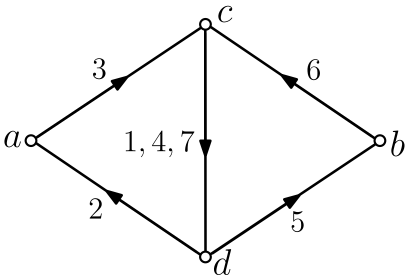
A strongly connected component (SCC) in a temporal graph is a set S of vertices such that, for every $u, v \in S$, there is a temporal path from u to v that uses only vertices from S.

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

Temporal strongly connected components

A further difference to the static case:

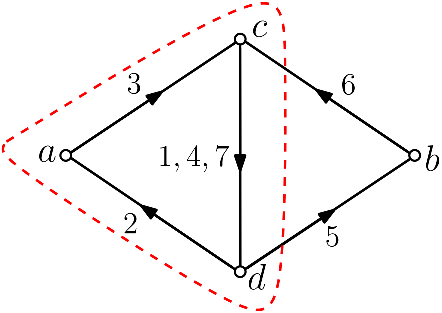
- two different SCCs can have common vertices



Temporal strongly connected components

A further difference to the static case:

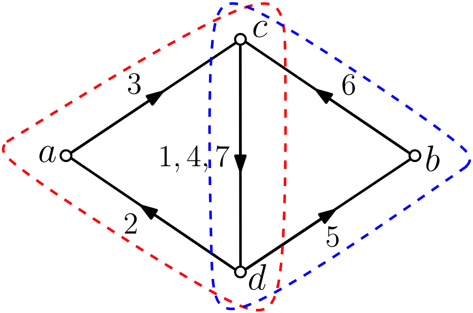
- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC



Temporal strongly connected components

A further difference to the static case:

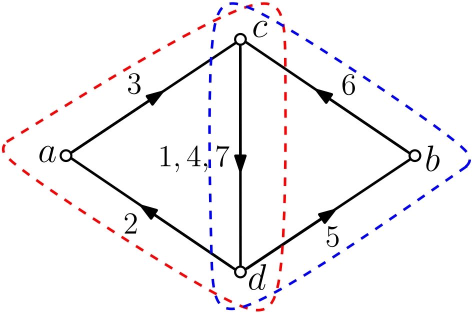
- two different SCCs can have common vertices
  - $\{a, c, d\}$ is a SCC
  - $\{b, c, d\}$ is another SCC



Temporal strongly connected components

A further difference 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

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: similarly to static graphs

Temporal strongly connected components

However:

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

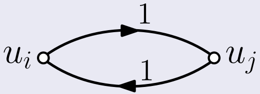
However:

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 (sketch):

- 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. $\square$

Overview

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

Menger's Theorem

Two fundamental duality results in (static) graph theory:

Theorem (Menger, 1927)

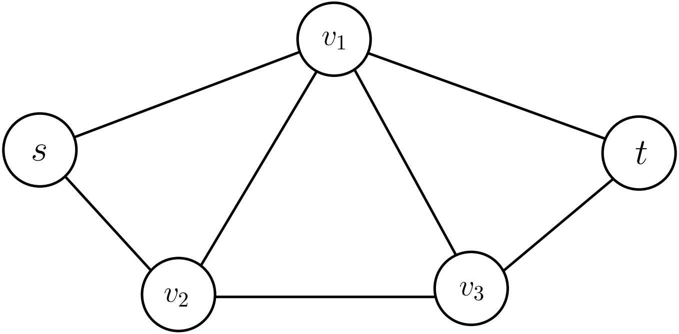
The maximum number of vertex-disjoint (edge-disjoint) s-t paths is equal to the minimum number of vertices (edges) needed to separate s from t.

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.

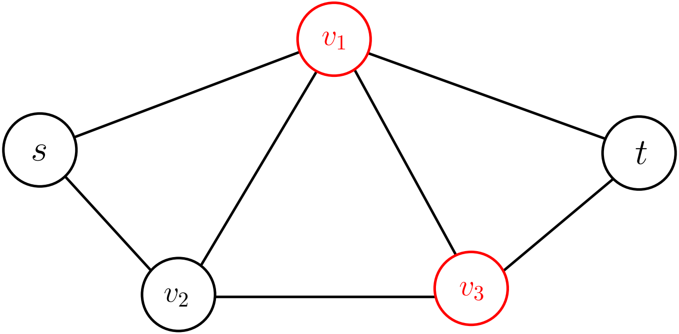


Menger's Theorem

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.

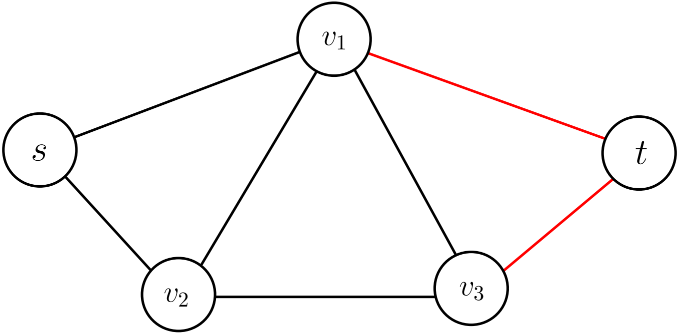


Menger's Theorem

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.

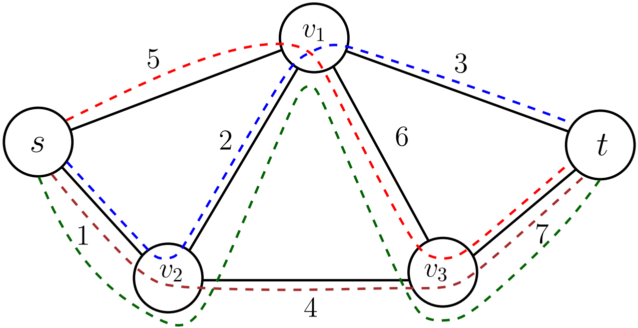


Menger's Theorem

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.

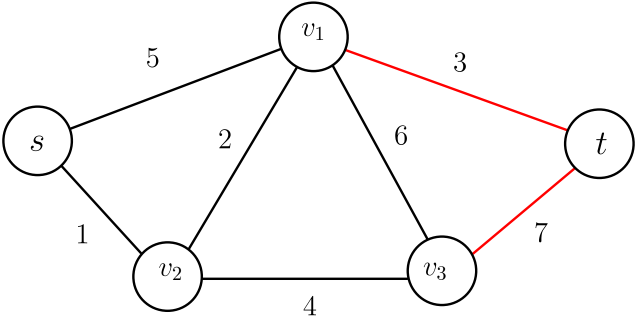


Menger's Theorem

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.

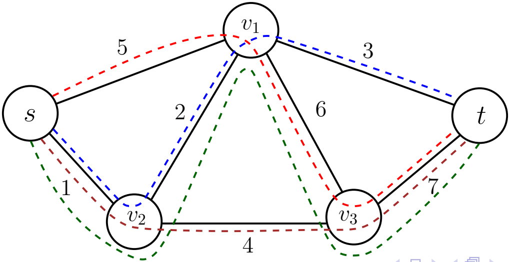
Menger's Theorem

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



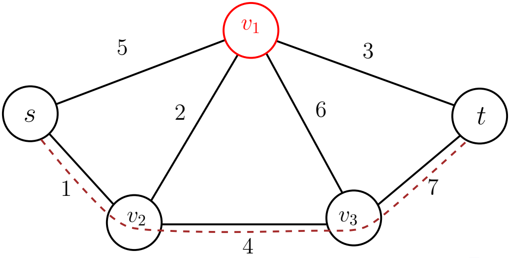
Menger's Theorem

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



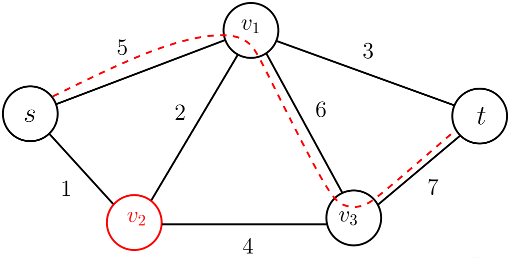
Menger's Theorem

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



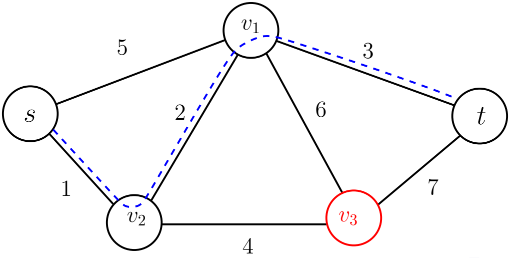
Menger's Theorem

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

An appropriate temporal vertex-version of Menger's theorem

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:

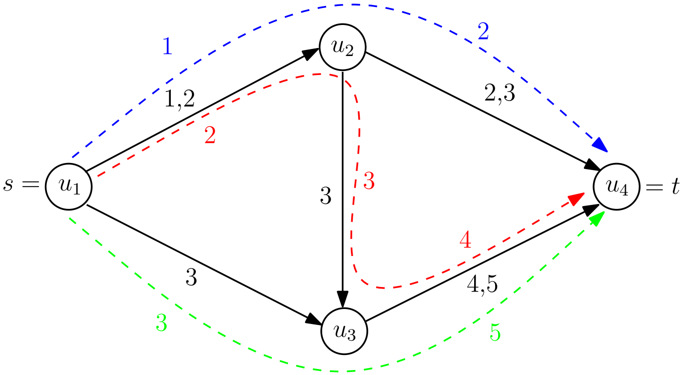
- two temporal paths are out-disjoint if they never leave from the same node at the same time
- we remove departure time t from vertex u if:
  - we remove label t from for all edges (u, w)

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:



Overview

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

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

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

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

and two fundamental connectivity properties:

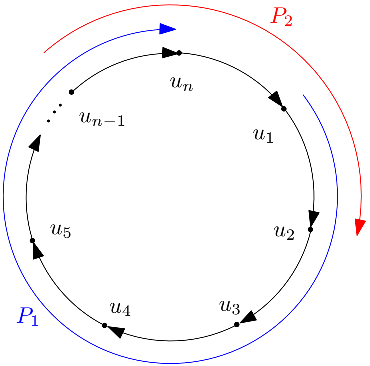
1. preserve in (G, λ) all reachabilities in G
   - if v is reachable from u in $G \Rightarrow u \rightsquigarrow v$ in (G, λ)
2. preserve in (G, λ) all paths in G
   - G has a path $P \Rightarrow (G, \lambda)$ has a temporal path on the same edges as P

Temporality of the ring C_n

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

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

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

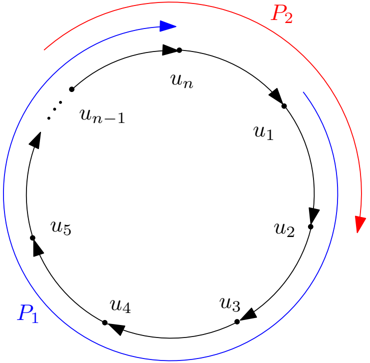


Temporality of the ring C_n

- The labeling assigning to each edge (u_i, u_{i+1}) the labels $\{i, n+i\}$ preserves all 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 $2n-2$)



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

- Assume that some edge e 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 $n-1$

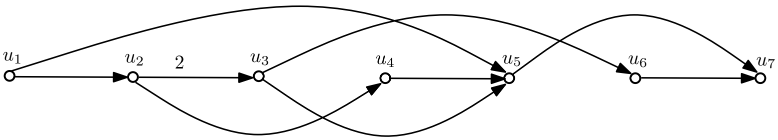
$\Rightarrow$ the optimal labeling assigns $\{1, 2, , \ldots, n-1\}$ to all edges of C_n

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

Temporality of a DAG

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

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



Temporality of a DAG

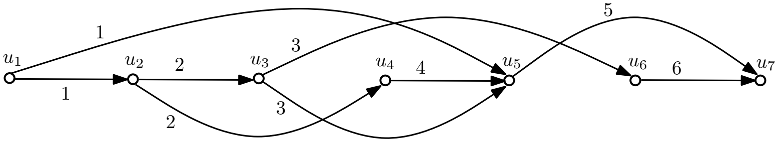
Lemma (Mertzios, Michail, Chatzigiannakis, Spirakis, *ICALP*, 2013)

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 label i to every edge (u_i, u_j), where $i < j$.

$\square$



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

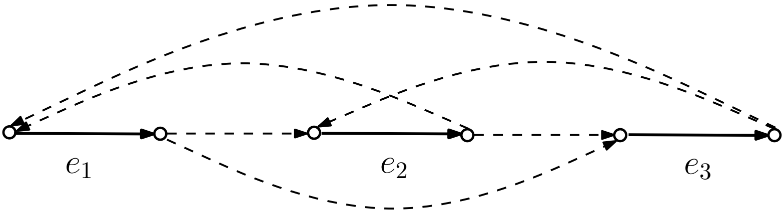
Intuition gained:

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

Based on this intuition:

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

The set $S = \{e_1, e_2, \ldots, e_k\} \subseteq E(G)$ is an edge-kernel of G if for every permutation of S there is a (static) path of G that visits all edges of S 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 $\tau(G, \text{all paths}) \geq k$.

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

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

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

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 $\Omega(n^{\frac{1}{3}})$.

Temporality: preserving all reachabilities

If we want to preserve only the 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 idea:

- pick an arbitrary vertex v (as the "root")
- build a temporal in-tree to vertex v
- from v build a temporal out-tree to vertex v

Temporality: preserving all reachabilities

If we want to preserve only the 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 idea:

- pick an arbitrary vertex v (as the "root")
- build a temporal in-tree to vertex v
- from v build a temporal out-tree to vertex v

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.

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.

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.

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.

Overview

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

Research Directions

- Find constant-factor approximations for the various temporal graph design problems
- 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]
    [Himmel, Molter, Niedermeier, Sorge, *Social Network Analysis and Mining*, 2017]
- 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?

Research Directions

New EPSRC project
(Starting September 2017):

Title: Algorithmic Aspects of Temporal Graphs

P.I. in Durham: George Mertzios

- PostDoc: Viktor Zamaraev

P.I. in Liverpool: Paul Spirakis

- PostDoc: Eleni Akrida

Value total amount: £ 800,000

Thank you for your attention!