Temporal Graph Mining

Aristides Gionis, Lutz Oettershagen, Ilie Sarpe

{argioni, lutzo, ilsarpe}@kth.se

KTH Royal Institute of Technology

Tutors

Aristides Gionis

Lutz Oettershagen

Ilie Sarpe

Website

https://miningtemporalnetworks.github.io/

Interaction

`menti.com`

code: 14 46 97 6

Agenda

Part I : **Introduction and Motivation**

- models of temporal networks
- algorithmic approaches

Part II : **Mining Temporal Networks A:**

- connectivity, temporal properties
- centrality, cores

Part III : **Mining Temporal Networks B:**

- communities, patterns and events
- diffusion and random networks

Part IV : **Tools and Code Libraries**

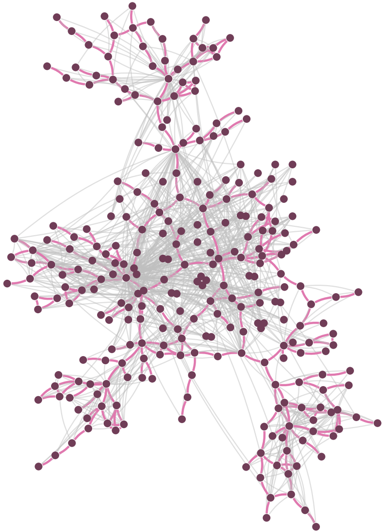
Part V : **Challenges, Open Problems, and Trends**

Part I

Introduction and Motivation

Interconnected World

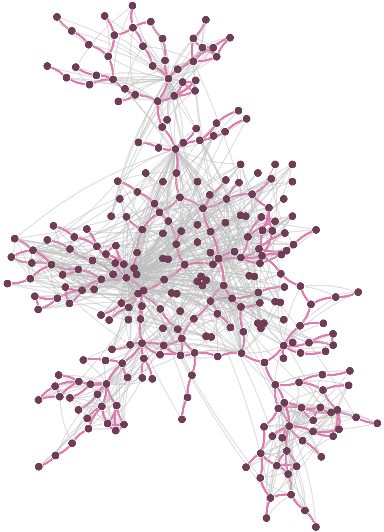
- networks model objects and their relations



Interconnected World

- networks model objects and their relations

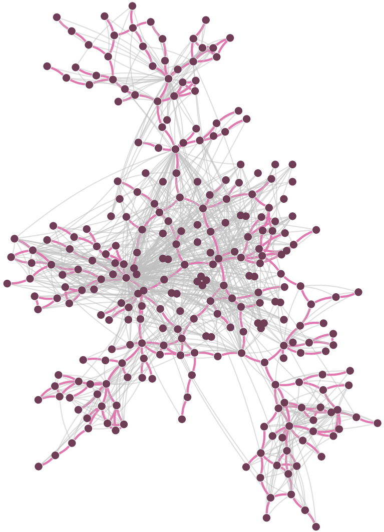
- many different network types
  - social (WhatsApp, LinkedIn, etc..)



Interconnected World

- networks model objects and their relations

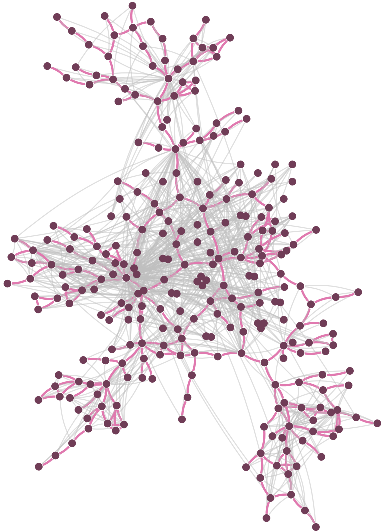
- many different network types
  - social (WhatsApp, LinkedIn, etc..)
  - informational (co-authorship, email, etc..)



Interconnected World

- networks model objects and their relations

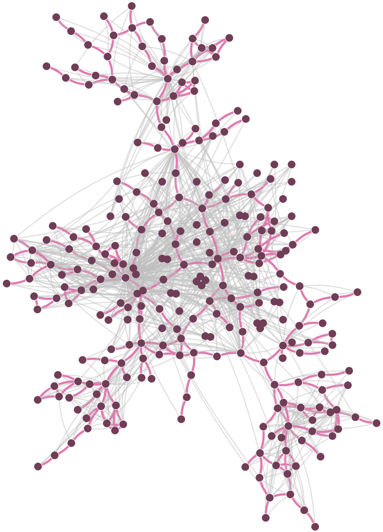
- many different network types
  - social (WhatsApp, LinkedIn, etc..)
  - informational (co-authorship, email, etc..)
  - technological (IP-level, transportation, etc..)



Interconnected World

- networks model objects and their relations

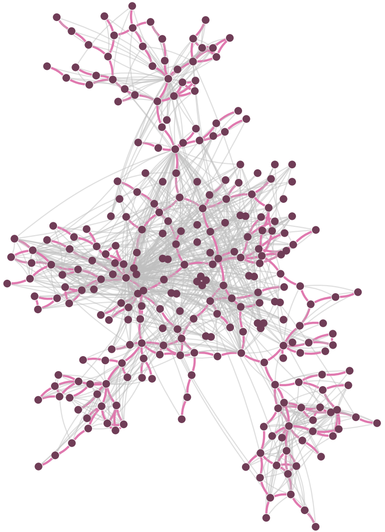
- many different network types
  - social (WhatsApp, LinkedIn, etc..)
  - informational (co-authorship, email, etc..)
  - technological (IP-level, transportation, etc..)
  - purchase (E-commerce, crypto, etc..)



Interconnected World

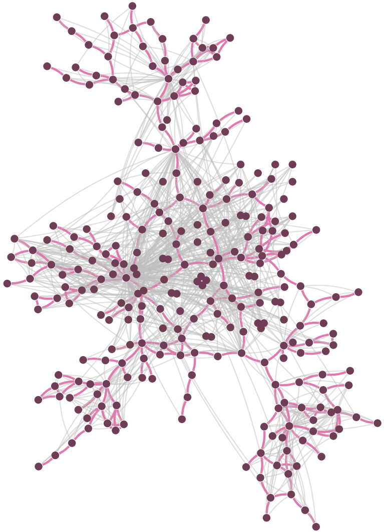
- networks model objects and their relations

- many different network types
  - social (WhatsApp, LinkedIn, etc..)
  - informational (co-authorship, email, etc..)
  - technological (IP-level, transportation, etc..)
  - purchase (E-commerce, crypto, etc..)
  - ...



Interconnected World

- networks model objects and their relations
- many different network types
  - social (WhatsApp, LinkedIn, etc..)
  - informational (co-authorship, email, etc..)
  - technological (IP-level, transportation, etc..)
  - purchase (E-commerce, crypto, etc..)
  - ...
- networks are ubiquitous in WWW-based applications



Impact of Networks

- online communication networks and social media

Impact of Networks

- online communication networks and social media
- implications in
  - knowledge creation

Impact of Networks

- online communication networks and social media

- implications in
  - knowledge creation
  - peoples' behavior

Impact of Networks

- online communication networks and social media

- implications in
  - knowledge creation
  - peoples' behavior
  - information sharing

Impact of Networks

- online communication networks and social media

- implications in
  - knowledge creation
  - peoples' behavior
  - information sharing
  - education

Impact of Networks

- online communication networks and social media
- implications in
  - knowledge creation
  - peoples' behavior
  - information sharing
  - education
  - democracy

Impact of Networks

- online communication networks and social media

- implications in
  - knowledge creation
  - peoples' behavior
  - information sharing
  - education
  - democracy
  - society as a whole

Impact of Networks

- 4.76 billion social media users,
  ca. 60% of the world's population

Impact of Networks

- 4.76 billion social media users,
  ca. 60% of the world's population
- 73.3% of all global spending on advertising for digital platforms in 2022

Impact of Networks

- 4.76 billion social media users,
  ca. 60% of the world's population
- 73.3% of all global spending on advertising for digital platforms in 2022
- 53.9% of users are concerned about misinformation

Impact of Networks

- 4.76 billion social media users,
  ca. 60% of the world's population
- 73.3% of all global spending on advertising
  for digital platforms in 2022
- 53.9% of users are concerned about misinformation
- insights can lead to huge monetary and societal impacts

Research Questions in Network Mining

- structure discovery
  - communities, summarization, events, role mining

Research Questions in Network Mining

- structure discovery
  - communities, summarization, events, role mining
- study complex dynamic phenomena
  - evolution, information diffusion, opinion formation, structural prediction, patterns

Research Questions in Network Mining

- structure discovery
  - communities, summarization, events, role mining
- study complex dynamic phenomena
  - evolution, information diffusion, opinion formation, structural prediction, patterns
- develop novel applications and mining primitives

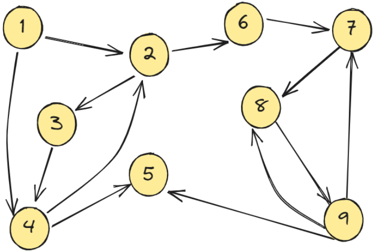
Research Questions in Network Mining

- structure discovery
  - communities, summarization, events, role mining
- study complex dynamic phenomena
  - evolution, information diffusion, opinion formation, structural prediction, patterns
- develop novel applications and mining primitives
- design efficient algorithms

Network Mining: Traditional View

- networks represented as pure graph-theory objects

  no additional vertex / edge information

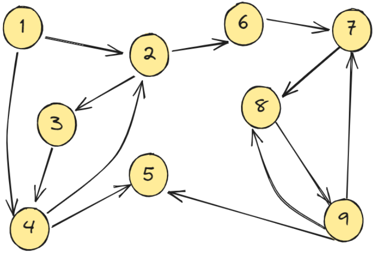


Network Mining: Traditional View

- networks represented as pure graph-theory objects

  no additional vertex / edge information

- emphasis on static networks



Temporal Networks: A new lens for network mining

- ability to collect and store large volumes of network data

Temporal Networks: A new lens for network mining

- ability to collect and store large volumes of network data
- available data have time granularity

Temporal Networks: A new lens for network mining

- ability to collect and store large volumes of network data
- available data have time granularity
- lots of additional information associated to vertices/edges

Temporal Networks: A new lens for network mining

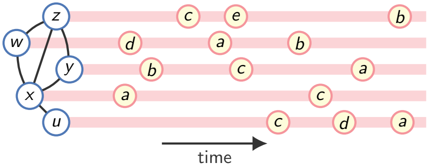
- ability to collect and store large volumes of network data
- available data have time granularity
- lots of additional information associated to vertices/edges
- capturing activity and interaction occurring over systems

Temporal Networks: A new lens for network mining

- ability to collect and store large volumes of network data
- available data have time granularity
- lots of additional information associated to vertices/edges
- capturing activity and interaction occurring over systems
- giving rise to new concepts, new problems, and new computational challenges and opportunities

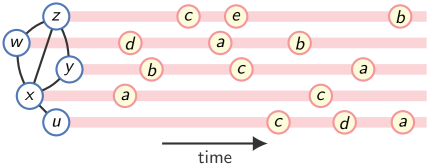
Modeling Activity in Networks

1. network nodes perform actions (e.g., posting messages)

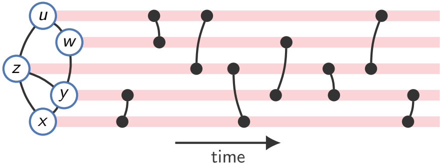


Modeling Activity in Networks

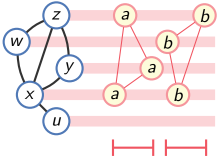
1. network nodes perform actions (e.g., posting messages)



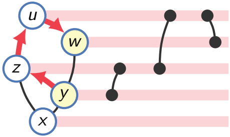
2. network nodes interact with each other
   (e.g., a "like", a repost, or sending a message to each other)



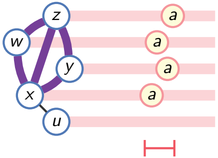
Many Novel and Interesting Concepts



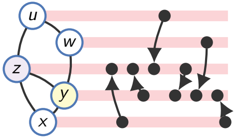
new pattern types



temporal information paths



new types of events



network evolution

Temporal Network Mining — Objectives

- identify new phenomena to be captured

Temporal Network Mining — Objectives

- identify new phenomena to be captured
- formulate suitable problems capturing the inherent complexity

Temporal Network Mining — Objectives

- identify new phenomena to be captured
- formulate suitable problems capturing the inherent complexity
- develop algorithmic approaches

Temporal Network Mining — Objectives

- identify new phenomena to be captured
- formulate suitable problems capturing the inherent complexity
- develop algorithmic approaches
- analyze real-world data and gain novel insights

Terminology

- we use term "temporal networks", but terminology is not standardized

Terminology

- we use term "temporal networks", but terminology is not standardized
- term "X Y" can be encountered in the literature, where

Terminology

- we use term “temporal networks”, but terminology is not standardized
- term “X Y” can be encountered in the literature, where

X :

temporal
dynamic
(time-)evolving
time-varying
time-dependent
evolutionary

Y :

networks
graphs

Terminology

- we use term "temporal networks", but terminology is not standardized
- term "X Y" can be encountered in the literature, where

X:

- temporal
- dynamic
- (time-)evolving
- time-varying
- time-dependent
- evolutionary

Y:

- networks
- graphs

- some combinations have distinct meaning, but not always

Examples of Temporal Networks

- online communication networks
  - phone, email, text messages, etc.

(Holme, 2015)

Examples of Temporal Networks

- online communication networks
  - phone, email, text messages, etc.
- economic networks
  - credit card transactions
  - trade networks of countries
  - bitcoin transcations

(Holme, 2015)

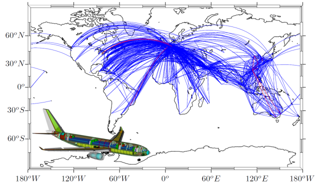
Examples of Temporal Networks

- online communication networks
  - phone, email, text messages, etc.
- economic networks
  - credit card transactions
  - trade networks of countries
  - bitcoin transcations
- bibliographic networks
  - collaboration and citation networks

(Holme, 2015)

Examples of Temporal Networks

- human proximity networks
  - recorded by various sensors and devices, e.g., bluetooth, wifi, etc.
  - patient-referral networks, i.e., how patients are transferred between wards of a hospital system
  - sexual contact networks

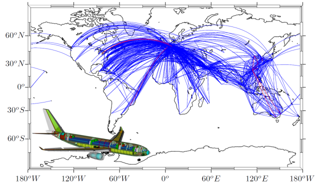


(Holme, 2015)

Examples of Temporal Networks

- human proximity networks
  - recorded by various sensors and devices, e.g., bluetooth, wifi, etc.
  - patient-referral networks, i.e., how patients are transferred between wards of a hospital system
  - sexual contact networks

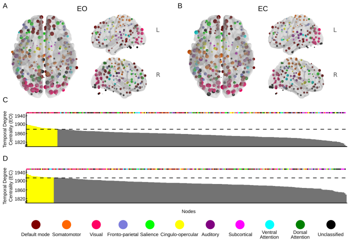
- travel and transportation networks
  - airline connections, bus transport, bike-sharing systems



(Holme, 2015)

Examples of Temporal Networks

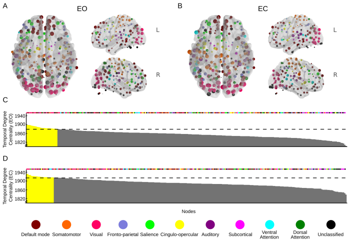
- brain networks
  - temporal correlations of the oxygen levels of brain regions as measured by fMRI scanning



(Holme, 2015)

Examples of Temporal Networks

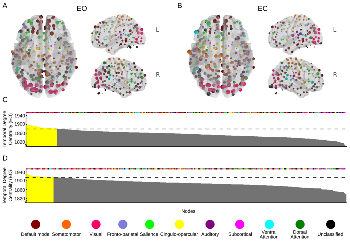
- brain networks
  - temporal correlations of the oxygen levels of brain regions as measured by fMRI scanning
- biological networks
  - genes involved in different interactions that change over time



(Holme, 2015)

Examples of Temporal Networks

- brain networks
  - temporal correlations of the oxygen levels of brain regions as measured by fMRI scanning
- biological networks
  - genes involved in different interactions that change over time
- animal proximity networks
  - obtained via RFID devices
  - lifestock or wildlife



(Holme, 2015)

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and
  - set of edges $E = \{(u, v, t)\}$, with $u, v \in V$ and $t \in T$ where T is the time domain

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and
  - set of edges $E = \{(u, v, t)\}$, with $u, v \in V$ and $t \in T$ where T is the time domain
  - T is usually $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and
  - set of edges $E = \{(u, v, t)\}$, with $u, v \in V$ and $t \in T$ where T is the time domain
  - T is usually $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$
  - if interactions have duration, then $E = \{(u, v, t, \lambda)\}$ with λ in $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$

Representation of Temporal Networks

1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and
  - set of edges $E = \{(u, v, t)\}$, with $u, v \in V$ and $t \in T$ where T is the time domain
  - T is usually $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$
  - if interactions have duration, then $E = \{(u, v, t, \lambda)\}$ with λ in $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$
- this is a lossless representation

Representation of Temporal Networks

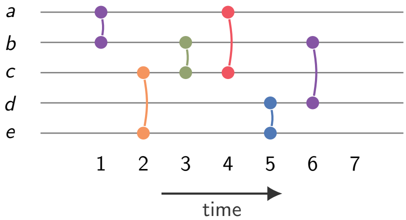
1. Sequence of interactions

- a temporal network is represented as $G = (V, E)$
  - with (static) set of nodes V, and
  - set of edges $E = \{(u, v, t)\}$, with $u, v \in V$ and $t \in T$ where T is the time domain
  - T is usually $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$
  - if interactions have duration, then $E = \{(u, v, t, \lambda)\}$ with λ in $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$
- this is a lossless representation
- also known as sequence of contacts, or sequence of (temporal) edges or temporal edge stream
- usually, edges given in chronological order

Representation of Temporal Networks

1. Sequence of interactions

- visual representation of a temporal network as a sequence of interactions



Representation of Temporal Networks

2. Sequence of static graphs

- sequence $G_1, \dots, G_T$

Representation of Temporal Networks

2. Sequence of static graphs

- sequence $G_1, \ldots, G_T$
  - where $G_t = (V_t, E_t)$, with $t = 1, \ldots, T$ with $T \in \mathbb{N}$

Representation of Temporal Networks

2. Sequence of static graphs

- sequence $G_1, \ldots, G_T$
  - where $G_t = (V_t, E_t)$, with $t = 1, \ldots, T$ with $T \in \mathbb{N}$
  - typically assume that nodes are fixed, i.e., $V_t = V$

Representation of Temporal Networks

2. Sequence of static graphs

- sequence $G_1, \ldots, G_T$
  - where $G_t = (V_t, E_t)$, with $t = 1, \ldots, T$ with $T \in \mathbb{N}$
  - typically assume that nodes are fixed, i.e., $V_t = V$
  - E_t are the edges that occur in time interval t

Representation of Temporal Networks

2. Sequence of static graphs

- sequence $G_1, \ldots, G_T$
  - where $G_t = (V_t, E_t)$, with $t = 1, \ldots, T$ with $T \in \mathbb{N}$
  - typically assume that nodes are fixed, i.e., $V_t = V$
  - E_t are the edges that occur in time interval t

- not equivalent representation with sequence of interactions

Representation of Temporal Networks

2. Sequence of static graphs

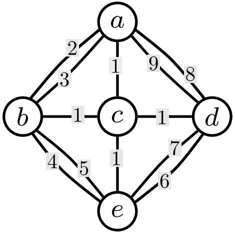
- sequence $G_1, \dots, G_T$
  - where $G_t = (V_t, E_t)$, with $t = 1, \dots, T$ with $T \in \mathbb{N}$
  - typically assume that nodes are fixed, i.e., $V_t = V$
  - E_t are the edges that occur in time interval t

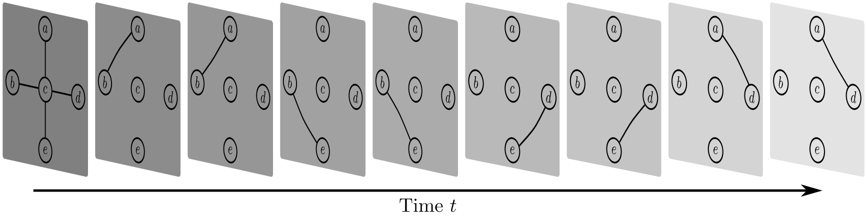
- not equivalent representation with sequence of interactions
  - representation depends on quantization parameter, e.g., seconds, minutes, hours, days, etc.

Representation of Temporal Networks

2. Sequence of static graphs

- visual representation of a temporal network as a sequence of static graphs





Time Granularity

- choosing the right time resolution is important

Time Granularity

- choosing the right time resolution is important
- quantization: binning of time stamps into time intervals of fixed size, e.g, seconds, minutes, hours, days

Time Granularity

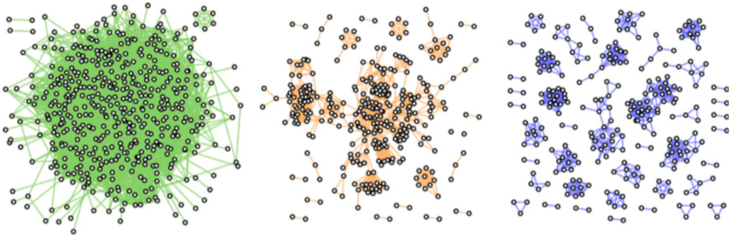
- choosing the right time resolution is important
- quantization: binning of time stamps into time intervals of fixed size, e.g, seconds, minutes, hours, days
  - coarse resolution may lead to information loss, dense time steps

Time Granularity

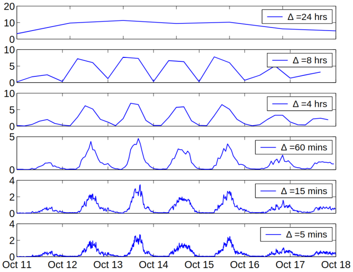
- choosing the right time resolution is important
- quantization: binning of time stamps into time intervals of fixed size, e.g, seconds, minutes, hours, days
  - coarse resolution may lead to information loss, dense time steps
  - fine resolution captures more information but may lead to sparse (or even empty) time steps

Time Granularity

- choosing the right time resolution is important
- quantization: binning of time stamps into time intervals of fixed size, e.g, seconds, minutes, hours, days
  - coarse resolution may lead to information loss, dense time steps
  - fine resolution captures more information but may lead to sparse (or even empty) time steps



time point in contact network with time resolution of 24h, 1h, and 5 minutes (Lehmann, 2019)



mean degrees for different time resolutions (Clauset and Eagle, 2012)

Representation of Temporal Networks

3. Time series of contacts

- a time-series for each pair of nodes in the network
- equivalent representation with sequence of interactions

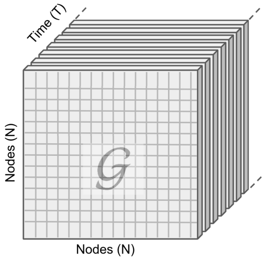
Representation of Temporal Networks

3. Time series of contacts

- a time-series for each pair of nodes in the network
- equivalent representation with sequence of interactions

4. Tensor representation

- tensor representing node $\times$ node $\times$ time information
- can apply powerful tensor-algebra techniques
- a complication is that time is directed, while tensor algebra assumes that indices can be relabeled (breaking the time ordering)
- equivalent representation with sequence of interactions



Representation of Temporal Networks

(Casteigts et al., 2012)

5. Time-varying graphs defined as $G = (V, E, T, p, \lambda)$, where
   - V : set of nodes
   - $E \subseteq V \times V$: set of edges
   - T : a time domain
   - $p : E \times T \to \{0, 1\}$: a presence function
   - $\lambda : E \times T \to \mathbb{R}$: a latency function

- equivalent representation with sequence of interactions

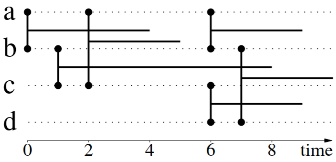
Representation of Temporal Networks

(Latapy et al., 2018)

6. Stream graphs and link streams

- a formalization for modeling interactions over time
- a stream graph is defined as $G = (T, V, W, E)$, where
  - T : a time domain
  - V : a set of nodes
  - $W \subseteq T \times V$: a set of temporal nodes
  - $E \subseteq T \times V \times V$: a set of links

    s.t., $(t, u, v) \in E$ implies $(t, u) \in W$ and $(t, v) \in W$
- stream graph: nodes are temporal too



- link stream equivalent representation with sequence of interactions

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$
  - and with $V_I = \{u, v \mid (u, v) \in E_I\}$

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$
  - and with $V_I = \{u, v \mid (u, v) \in E_I\}$

- time window can be complete time span of G

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$
  - and with $V_I = \{u, v \mid (u, v) \in E_I\}$
- time window can be complete time span of G
- possible to add weights to the static edges, reflecting different temporal aspects, e.g, number of contacts, or sum of durations of contacts

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$
  - and with $V_I = \{u, v \mid (u, v) \in E_I\}$

- time window can be complete time span of G
- possible to add weights to the static edges, reflecting different temporal aspects, e.g, number of contacts, or sum of durations of contacts

- not equivalent representation with sequence of interactions

Representation of Temporal Networks

(Holme, 2015)

7. Time window graph or underlying graph or projected graph

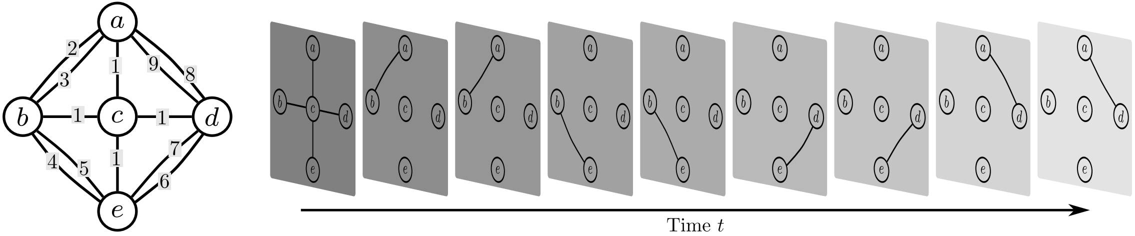
- static graph $G_I = (V_I, E_I)$ for given time interval (or window) $I = [a, b]$
  - with $E_I = \{(u, v) \mid (u, v, t) \in E \text{ and } t \in I\}$ be the set of edges with time stamp $t \in I$
  - and with $V_I = \{u, v \mid (u, v) \in E_I\}$

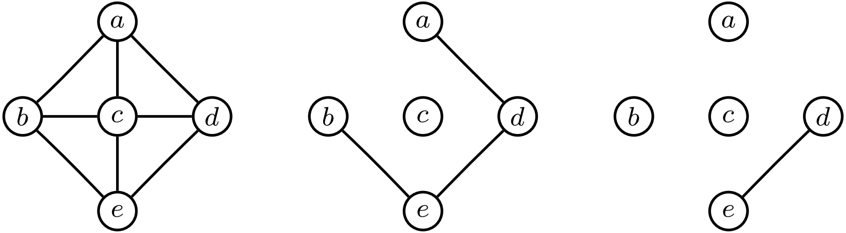
- time window can be complete time span of G
- possible to add weights to the static edges, reflecting different temporal aspects, e.g, number of contacts, or sum of durations of contacts

- not equivalent representation with sequence of interactions
- usually results in loss of information

Representation of Temporal Networks

7. time window graph or underlying graph or projected graph

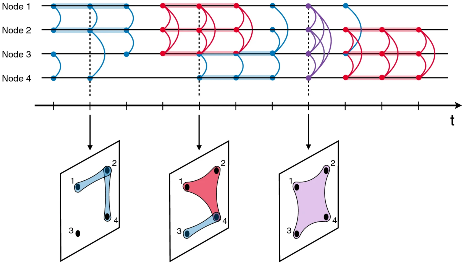




time window graphs for intervals $[1, 9]$, $[4, 9]$, $[6, 7]$

Temporal Graph Variants

- time-intervals instead of time stamps
- directed vs. undirected edges
- multi edges
- (time-variant) labeled or colored nodes and edges
- (time-variant) node and edge features
- temporal hypergraphs



(Cencetti et al., 2021)

Combinations possible: temporal multi-layer hypergraphs with node features

Temporal Networks vs. Dynamic Graphs

- dynamic graphs is a standard model typically studied in theoretical computer science
  - e.g., (Henzinger et al., 1999; Thorup, 2000; Hanauer et al., 2021)
- dynamic graphs are represented as a sequence of edge additions and/or edge deletions

Temporal Networks vs. Dynamic Graphs

- dynamic graphs is a standard model typically studied in theoretical computer science
  - e.g., (Henzinger et al., 1999; Thorup, 2000; Hanauer et al., 2021)
- dynamic graphs are represented as a sequence of edge additions and/or edge deletions
- objective: efficient maintenance of graph properties
  - e.g., connectivity, shortest paths, spanners, etc.

Temporal Networks vs. Dynamic Graphs

- dynamic graphs is a standard model typically studied in theoretical computer science
  - e.g., (Henzinger et al., 1999; Thorup, 2000; Hanauer et al., 2021)
- dynamic graphs are represented as a sequence of edge additions and/or edge deletions
- objective: efficient maintenance of graph properties
  - e.g., connectivity, shortest paths, spanners, etc.
- emphasis on computational efficiency
  - computation time per operation
  - e.g., cost of maintaining a minimum spanning tree per edge additions/deletions

Temporal Networks vs. Dynamic Graphs

- dynamic graphs resemble sequence of interactions model

Temporal Networks vs. Dynamic Graphs

- dynamic graphs resemble sequence of interactions model
- main difference lies on which graph properties we study
- for dynamic graphs we typically consider properties on graph snapshots
  - i.e., minimum spanning tree on the current snapshot

Temporal Networks vs. Dynamic Graphs

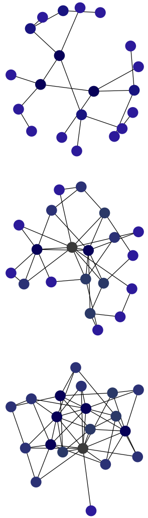
- dynamic graphs resemble sequence of interactions model
- main difference lies on which graph properties we study
- for dynamic graphs we typically consider properties on graph snapshots
  - i.e., minimum spanning tree on the current snapshot
- for temporal graphs we typically consider properties that span a time interval
  - i.e., a temporal pattern

Temporal Networks vs. Dynamic Graphs

- dynamic graphs resemble sequence of interactions model
- main difference lies on which graph properties we study
- for dynamic graphs we typically consider properties on graph snapshots
  - i.e., minimum spanning tree on the current snapshot
- for temporal graphs we typically consider properties that span a time interval
  - i.e., a temporal pattern
- disclaimer: we do not consider dynamic graphs

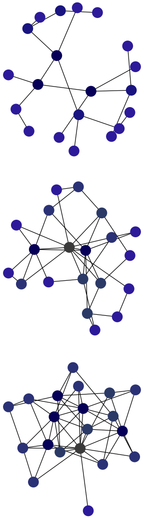
Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.



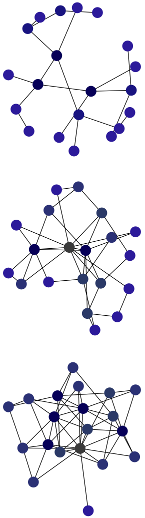
Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.
- similar to dynamic graphs: a sequence of node/edge additions (typically no deletions)



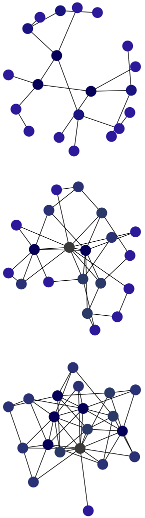
Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.
- similar to dynamic graphs: a sequence of node/edge additions (typically no deletions)
- node/edge addition are governed by a probabilistic model, not arbitrary, or worst case, as in algorithmic models



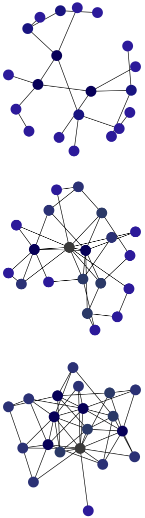
Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.
- similar to dynamic graphs: a sequence of node/edge additions (typically no deletions)
- node/edge addition are governed by a probabilistic model, not arbitrary, or worst case, as in algorithmic models
- emphasis again on network topology, i.e., how certain network structures emerge



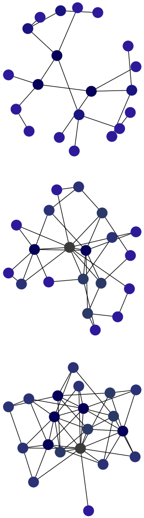
Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.
- similar to dynamic graphs: a sequence of node/edge additions (typically no deletions)
- node/edge addition are governed by a probabilistic model, not arbitrary, or worst case, as in algorithmic models
- emphasis again on network topology, i.e., how certain network structures emerge
  - e.g., scale-free distribution, small world, etc.



Dynamic Networks in Network Generation

- in graph generation models, we consider dynamic networks
  - e.g., Barabási-Albert, forest-fire, copying model, etc.
- similar to dynamic graphs: a sequence of node/edge additions (typically no deletions)
- node/edge addition are governed by a probabilistic model, not arbitrary, or worst case, as in algorithmic models
- emphasis again on network topology, i.e., how certain network structures emerge
  - e.g., scale-free distribution, small world, etc.
- disclaimer: we do not consider dynamic networks in this context



Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic
  - assume fast computation per data item processed, e.g., constant or poly-logarithmic

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic
  - assume fast computation per data item processed, e.g., constant or poly-logarithmic
- a graph stream is a graph dataset in the data-stream model

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic
  - assume fast computation per data item processed, e.g., constant or poly-logarithmic
- a graph stream is a graph dataset in the data-stream model
  e.g., sequence of interactions (temporal network), or edge additions/deletions (dynamic graph)

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic
  - assume fast computation per data item processed, e.g., constant or poly-logarithmic
- a graph stream is a graph dataset in the data-stream model
  e.g., sequence of interactions (temporal network), or edge additions/deletions (dynamic graph)
- a graph stream is not a representation model, but underlying computational model

Graph Streams

- data-stream model: (Muthukrishnan et al., 2005)
  - data are presented as a sequence of data items (potentially infinite)
  - assume a small number of passes, typically constant or just one pass
  - assume small memory compared to data size, e.g., poly-logarithmic
  - assume fast computation per data item processed, e.g., constant or poly-logarithmic
- a graph stream is a graph dataset in the data-stream model
  e.g., sequence of interactions (temporal network), or edge additions/deletions (dynamic graph)
- a graph stream is not a representation model, but underlying computational model
- we can study questions of mining temporal networks in the graph-stream model

Temporal Graph Learning

- rich and fast growing body of works on temporal graph learning

Temporal Graph Learning

- rich and fast growing body of works on temporal graph learning
- tasks: dynamic link/node property prediction, graph classification, clustering, link prediction, representation learning, ...

Temporal Graph Learning

- rich and fast growing body of works on temporal graph learning
- tasks: dynamic link/node property prediction, graph classification, clustering, link prediction, representation learning, ...
- methods: graph neural networks, recurrent neural network, graph transformers, ...

Temporal Graph Learning

- rich and fast growing body of works on temporal graph learning
- tasks: dynamic link/node property prediction, graph classification, clustering, link prediction, representation learning, ...
- methods: graph neural networks, recurrent neural network, graph transformers, ...
- several extensive surveys available, e.g.,
  (Gupta and Bedathur, 2022; Longa et al., 2023; Kazemi et al., 2020)

Temporal Graph Learning

- rich and fast growing body of works on temporal graph learning
- tasks: dynamic link/node property prediction, graph classification, clustering, link prediction, representation learning, ...
- methods: graph neural networks, recurrent neural network, graph transformers, ...
- several extensive surveys available, e.g.,
  (Gupta and Bedathur, 2022; Longa et al., 2023; Kazemi et al., 2020)
- disclaimer: in this tutorial we do not consider temporal graph learning

Theoretical Aspects of Temporal Graphs

- how is the complexity of classic combinatorial optimization problems changes when time is added?
- some old results: the max-flow min-cut theorem holds with unit capacities for temporal paths (Berman, 1996)

Theoretical Aspects of Temporal Graphs

- how is the complexity of classic combinatorial optimization problems changes when time is added?
- some old results: the max-flow min-cut theorem holds with unit capacities for temporal paths (Berman, 1996)
- a number of recent works
  - graph coloring (Mertzios et al., 2018)
  - maximal matching (Mertzios et al., 2019)
  - cliques (Viard et al., 2015, 2016; Himmel et al., 2017; Mertzios et al., 2024)
  - network design (Akrida et al., 2017; Enright et al., 2021)
  - path problems (Casteigts et al., 2021; Klobas et al., 2022, 2023)
  - vertex cover (Hamm et al., 2022; Akrida et al., 2020)
  - ...
- discussing complexity, FPT algorithms, enumeration, etc.

Agenda

Part I : **Introduction and Motivation**

- models of temporal networks
- algorithmic approaches

Part II : **Mining Temporal Networks A:**

- connectivity, temporal properties
- centrality, cores

Part III : **Mining Temporal Networks B:**

- communities, patterns and events
- diffusion and random networks

Part IV : **Tools and Code Libraries**

Part V : **Challenges, Open Problems, and Trends**

Part II

Mining Temporal Networks A

Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks

Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that

Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that
  - consecutive edges share a common node, and

Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that
  - consecutive edges share a common node, and
  - time stamps of temporal edges are increasing

Time-respecting Walks and Paths

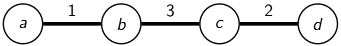
- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that
  - consecutive edges share a common node, and
  - time stamps of temporal edges are increasing
  - (non-strict version: time stamps non-decreasing)

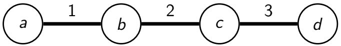
Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that
  - consecutive edges share a common node, and
  - time stamps of temporal edges are increasing
  - (non-strict version: time stamps non-decreasing)
- a temporal paths is a temporal walk that visits each vertex at most once

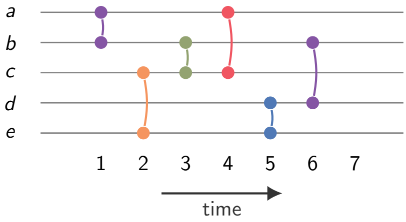
Time-respecting Walks and Paths

- a fundamental concept in analysis of temporal networks
- a time-respecting (or temporal) walk is a sequence of temporal edges, such that
  - consecutive edges share a common node, and
  - time stamps of temporal edges are increasing
  - (non-strict version: time stamps non-decreasing)
- a temporal paths is a temporal walk that visits each vertex at most once



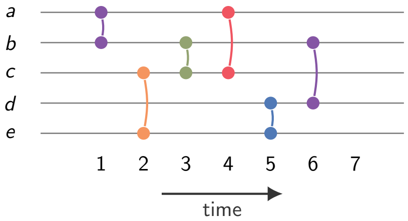


Time-respecting Paths — Example



$(c, e, 2), (e, d, 5), (d, b, 6)$ is a time-respecting path from c to b

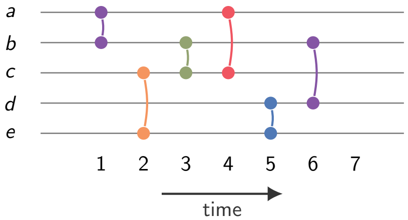
Time-respecting Paths — Example



$(c, e, 2), (e, d, 5), (d, b, 6)$ is a time-respecting path from c to b

$(c, b, 3), (b, a, 1)$ is not a time-respecting path

Time-respecting Paths — Example

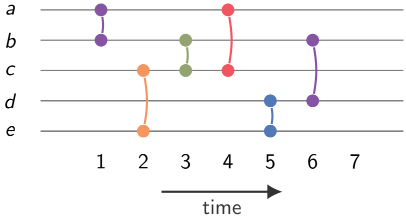


$(c, e, 2), (e, d, 5), (d, b, 6)$ is a time-respecting path from c to b

$(c, b, 3), (b, a, 1)$ is not a time-respecting path

- non-symmetric: from e to b but not from b to e

Time-respecting Paths — Example



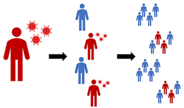
$(c, e, 2), (e, d, 5), (d, b, 6)$ is a time-respecting path from c to b

$(c, b, 3), (b, a, 1)$ is not a time-respecting path

- non-symmetric: from e to b but not from b to e
- non-transitive: from b to d and from d to e but not from b to e

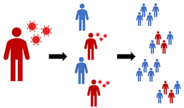
Applications

- information (or disease) can only propagate over time-respecting walks



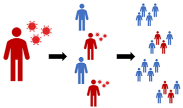
Applications

- information (or disease) can only propagate over time-respecting walks
- communication networks: capture possible flow of information



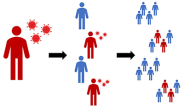
Applications

- information (or disease) can only propagate over time-respecting walks
- communication networks: capture possible flow of information
- financial networks: trace the sequence of financial exchanges to identify patterns, detect fraudulent activity, or assess market dynamics



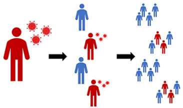
Applications

- information (or disease) can only propagate over time-respecting walks
- communication networks: capture possible flow of information
- financial networks: trace the sequence of financial exchanges to identify patterns, detect fraudulent activity, or assess market dynamics
- epidemiology: understanding the spread of diseases



Applications

- information (or disease) can only propagate over time-respecting walks
- communication networks: capture possible flow of information
- financial networks: trace the sequence of financial exchanges to identify patterns, detect fraudulent activity, or assess market dynamics
- epidemiology: understanding the spread of diseases



- social network analysis: centrality measures for ranking users

Temporal Reachability

Reachability is defined as in static graphs, but using time-respecting walks

Temporal Reachability

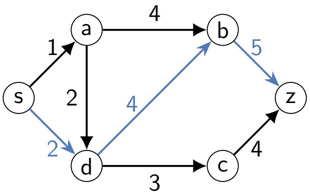
Reachability is defined as in static graphs, but using time-respecting walks

"There exists temporal (s, z)-walk in G" not always useful—the existence depends on time interval

Temporal Reachability

Reachability is defined as in static graphs, but using time-respecting walks

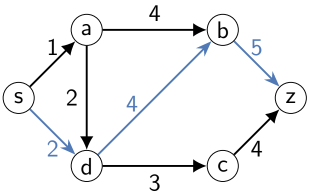
"There exists temporal (s, z)-walk in G" not always useful—the existence depends on time interval



Temporal Reachability

Reachability is defined as in static graphs, but using time-respecting walks

"There exists temporal (s, z)-walk in G" not always useful—the existence depends on time interval

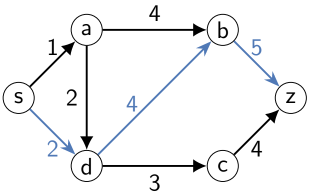


- temporal walks and paths problems, and reachability are always related to a time interval I

Temporal Reachability

Reachability is defined as in static graphs, but using time-respecting walks

"There exists temporal (s, z)-walk in G" not always useful—the existence depends on time interval

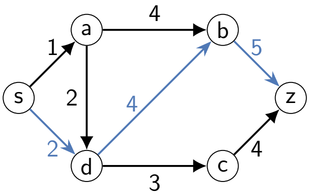


- temporal walks and paths problems, and reachability are always related to a time interval I
- if no time-interval given, we take the complete span of G

Temporal Reachability

Reachability is defined as in static graphs, but using time-respecting walks

"There exists temporal (s, z)-walk in G" not always useful—the existence depends on time interval



- temporal walks and paths problems, and reachability are always related to a time interval I
- if no time-interval given, we take the complete span of G
- finding all from s reachable nodes in linear time (later)

A Reachability Problem

Temporal Exploration Problem

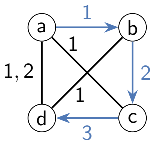
- **Given**: Temporal graph $G = (V, E)$, vertex $s \in V$
- **Question**: Can we reach all other nodes with a single temporal walk starting from s?

A Reachability Problem

Temporal Exploration Problem

- **Given**: Temporal graph $G = (V, E)$, vertex $s \in V$
- **Question**: Can we reach all other nodes with a single temporal walk starting from s?

Problem is NP-complete (corresponding problem in static graphs in linear time!)



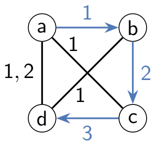
If transition time non-zero $\Rightarrow$ each edge at most once in a strict temporal walk

A Reachability Problem

Temporal Exploration Problem

- **Given**: Temporal graph $G = (V, E)$, vertex $s \in V$
- **Question**: Can we reach all other nodes with a single temporal walk starting from s?

Problem is NP-complete (corresponding problem in static graphs in linear time!)



If transition time non-zero $\Rightarrow$ each edge at most once in a strict temporal walk

Proof idea:

- reduction from Hamilton path problem

(Michail and Spirakis, 2016)

Connectivity and Connected Components

- static connected components are based on reachability
  - each node in connected component C can reach each all other nodes in C
- equivalence relation that partitions the graph
  - reflexivity, symmetry, transitivity

Connectivity and Connected Components

- static connected components are based on reachability
  - each node in connected component C can reach each all other nodes in C
- equivalence relation that partitions the graph
  - reflexivity, symmetry, transitivity

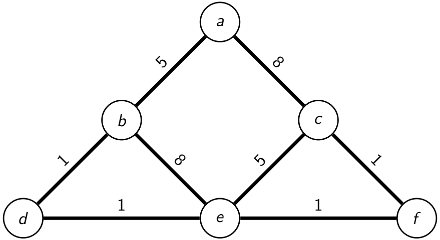
- temporal connected components are based on temporal reachability
  - a subset of the nodes $C \subseteq V$
  - there is a temporal walk between each pair $u, v \in C$

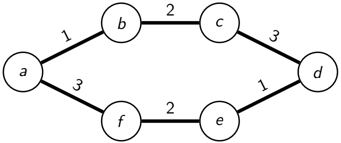
Connectivity and Connected Components

Temporal Connectivity Problem

- **Given:** A temporal graph and integer k
- **Question:** Is there a subset of the vertices $V' \subseteq V$ of size k such that all vertices in V' can reach each other by a temporal path?

- two versions: open variant allows paths using nodes outsides of V', closed variant not
- both cases are NP-complete (Bhadra and Ferreira, 2003)



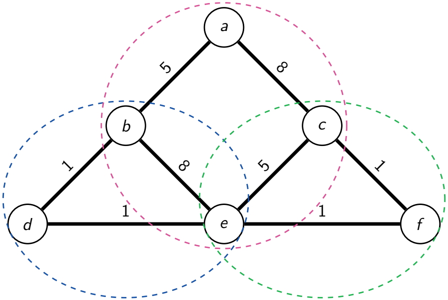


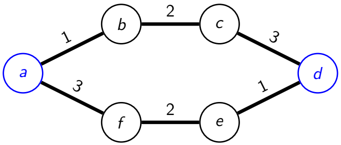
Connectivity and Connected Components

Temporal Connectivity Problem

- **Given:** A temporal graph and integer k
- **Question:** Is there a subset of the vertices $V' \subseteq V$ of size k such that all vertices in V' can reach each other by a temporal path?

- two versions: open variant allows paths using nodes outsides of V', closed variant not
- both cases are NP-complete (Bhadra and Ferreira, 2003)





a and d are openly connected

Connected components can be overlapping

Time Window Reachability and Connectivity

- compute reachability or connected components in time window graph $G_I = (V_I, E_I)$
  - given time window $I = [a, b]$, $G_I = (V_I, E_I)$ is static graph induced by edges appearing in I

Time Window Reachability and Connectivity

- compute reachability or connected components in time window graph $G_I = (V_I, E_I)$
  - given time window $I = [a, b]$, $G_I = (V_I, E_I)$ is static graph induced by edges appearing in I
- time window reachability for $u \in V$ in time interval $I = [a, b]$ (Wen et al., 2020)
  - the set of nodes that u can reach in $G_I = (V_I, E_I)$ with static walk

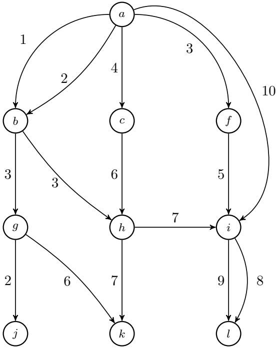
Time Window Reachability and Connectivity

- compute reachability or connected components in time window graph $G_I = (V_I, E_I)$
  - given time window $I = [a, b]$, $G_I = (V_I, E_I)$ is static graph induced by edges appearing in I
- time window reachability for $u \in V$ in time interval $I = [a, b]$ (Wen et al., 2020)
  - the set of nodes that u can reach in $G_I = (V_I, E_I)$ with static walk
- time window connected components in time interval $I = [a, b]$ (Xie et al., 2023)
  - the set of static connected components in $G_I = (V_I, E_I)$

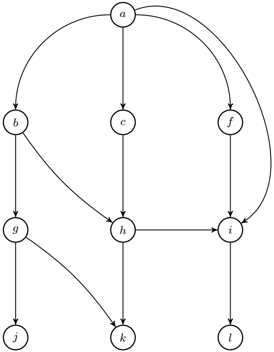
Time Window Reachability and Connectivity

- compute reachability or connected components in time window graph $G_I = (V_I, E_I)$
  - given time window $I = [a, b]$, $G_I = (V_I, E_I)$ is static graph induced by edges appearing in I
- time window reachability for $u \in V$ in time interval $I = [a, b]$ (Wen et al., 2020)
  - the set of nodes that u can reach in $G_I = (V_I, E_I)$ with static walk
- time window connected components in time interval $I = [a, b]$ (Xie et al., 2023)
  - the set of static connected components in $G_I = (V_I, E_I)$
- both problems solvable in linear time
- alternative: index-based algorithms for large scale graphs

Time-Respecting Paths



(a) Temporal Graph

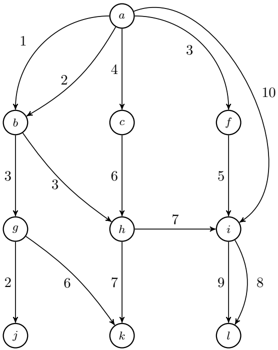


(b) Static Graph

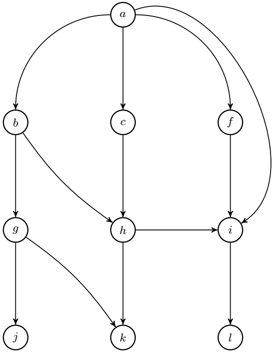
- some paths in the static graph are not meaningful in the temporal graph

(Wu et al., 2014)

Time-Respecting Paths



(a) Temporal Graph

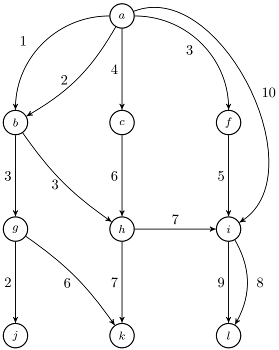


(b) Static Graph

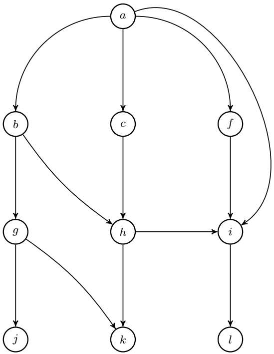
- some paths in the static graph are not meaningful in the temporal graph
- e.g., $a - b - g - j$ is not time-respecting path

(Wu et al., 2014)

Time-Respecting Paths



(a) Temporal Graph



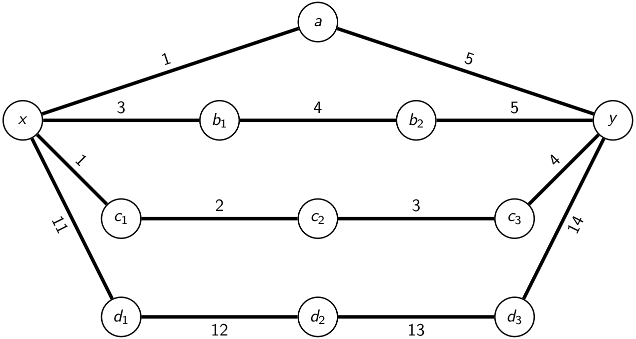
(b) Static Graph

- some paths in the static graph are not meaningful in the temporal graph
- e.g., $a-b-g-j$ is not time-respecting path
- what is an optimal path from a to k?

(Wu et al., 2014)

Minimum Temporal Paths

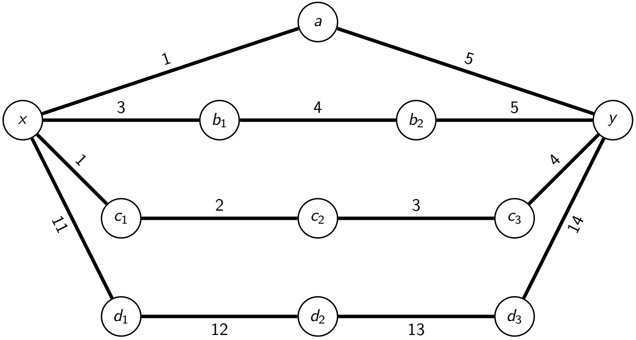
- earliest-arrival path : a path from x to y with earliest arrival time



(Wu et al., 2014)

Minimum Temporal Paths

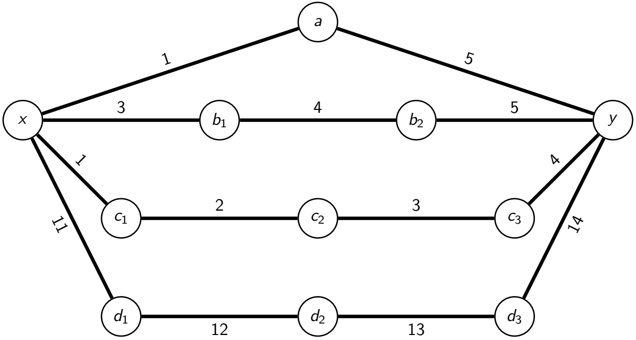
- earliest-arrival path: a path from x to y with earliest arrival time
- latest-departure path: a path from x to y with latest departure time



(Wu et al., 2014)

Minimum Temporal Paths

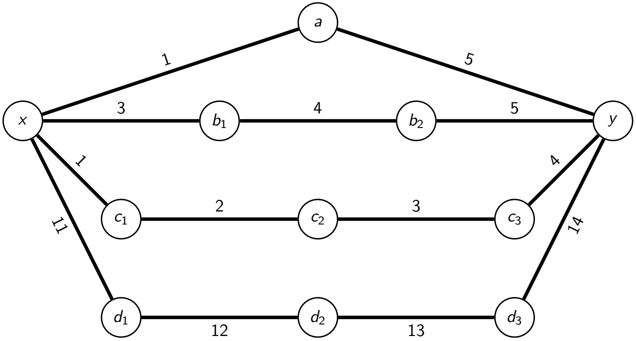
- earliest-arrival path : a path from x to y with earliest arrival time
- latest-departure path : a path from x to y with latest departure time
- fastest path : path from x to y with minimum elapsed time



(Wu et al., 2014)

Minimum Temporal Paths

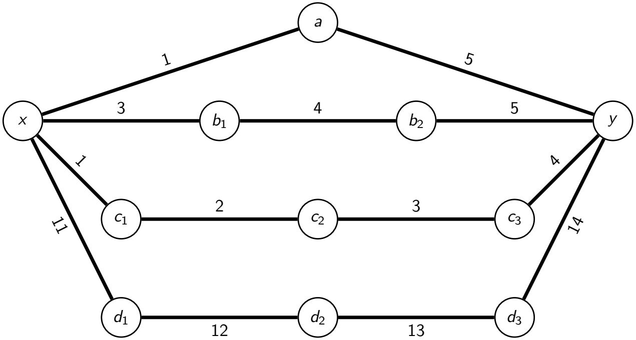
- earliest-arrival path : a path from x to y with earliest arrival time
- latest-departure path : a path from x to y with latest departure time
- fastest path : path from x to y with minimum elapsed time
- shortest path : a path from x to y minimum sum of traversal times



(Wu et al., 2014)

Minimum Temporal Paths

- earliest-arrival path : a path from x to y with earliest arrival time
- latest-departure path : a path from x to y with latest departure time
- fastest path : path from x to y with minimum elapsed time
- shortest path : a path from x to y minimum sum of traversal times
- minimum hop path: a path from x to y with minimum number of hops

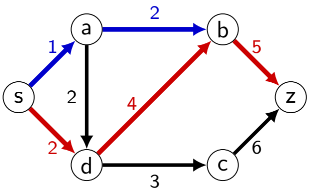


(Wu et al., 2014)

Optimal Path Computation

Observation

Let $P_{(s,z)}$ be an optimal temporal path and P a subpath of $P_{(s,z)}$, then, in general, P is not optimal



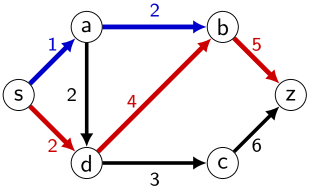
Example:

- fastest path (s, d, b, z) with duration 4

Optimal Path Computation

Observation

Let $P_{(s,z)}$ be an optimal temporal path and P a subpath of $P_{(s,z)}$, then, in general, P is not optimal



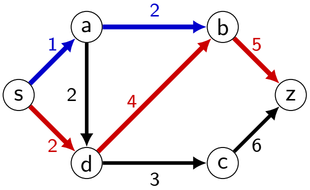
Example:

- fastest path (s, d, b, z) with duration 4
- subpath (s, d, b) has duration 3 but path (s, a, b) has duration 2

Optimal Path Computation

Observation

Let $P_{(s,z)}$ be an optimal temporal path and P a subpath of $P_{(s,z)}$, then, in general, P is not optimal



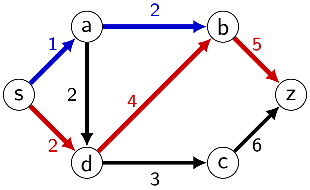
Example:

- fastest path (s, d, b, z) with duration 4
- subpath (s, d, b) has duration 3 but path (s, a, b) has duration 2
- similar examples for other variants

Optimal Path Computation

Observation

Let $P_{(s,z)}$ be an optimal temporal path and P a subpath of $P_{(s,z)}$, then, in general, P is not optimal



Example:

- fastest path (s, d, b, z) with duration 4
- subpath (s, d, b) has duration 3 but path (s, a, b) has duration 2
- similar examples for other variants
- greedy Dijkstra does not work in general

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source
- process edges (u, v, t, λ) in temporal order

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source
- process edges (u, v, t, λ) in temporal order
  - if $t \geq A[u]$ $\quad$ (u is already reached from x)

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source
- process edges (u, v, t, λ) in temporal order
  - if $t \geq A[u]$ (u is already reached from x)
  - current edge might be earliest path from x to v

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source
- process edges (u, v, t, λ) in temporal order
  - if $t \geq A[u]$ (u is already reached from x)
  - current edge might be earliest path from x to v
  - update $A[v] = \min(A[v], t + \lambda)$

Earliest-arrival Path

(Wu et al., 2014)

Subpath Optimality

If there exists an earliest arrival (s, z)-path, then there exist an earliest arrival (s, z)-path P such that each prefix path of P is an earliest arrival path

- source vertex x, starting time t_s
- array A of size $|V|$ to record arrival times to each node
- $A[x] = t_s$ and $A[v] = \infty$, for nodes other than source
- process edges (u, v, t, λ) in temporal order
  - if $t \geq A[u]$ (u is already reached from x)
  - current edge might be earliest path from x to v
  - update $A[v] = \min(A[v], t + \lambda)$

Linear time algorithm

Latest-departure Path

(Wu et al., 2014)

- temporal graph $G = (V, E)$
- sink vertex x, ending time t_s

Latest-departure Path

(Wu et al., 2014)

- temporal graph $G = (V, E)$
- sink vertex x, ending time t_s
- same process as for earliest-arrival path, but

Latest-departure Path

(Wu et al., 2014)

- temporal graph $G = (V, E)$
- sink vertex x, ending time t_s
- same process as for earliest-arrival path, but
- process edges in reversed temporal order

Latest-departure Path

(Wu et al., 2014)

- temporal graph $G = (V, E)$
- sink vertex x, ending time t_s
- same process as for earliest-arrival path, but
- process edges in reversed temporal order
- add new interaction to the path if it does not violate temporal order

Dominating Path

(Wu et al., 2014)

- source vertex x and sink v
- for a path P_1 arriving at v let (a, s), where
  - a : time of arrival at v
  - s : time of departure from x
- consider another path P_2 arriving at v with (a', s')

Dominating Path

(Wu et al., 2014)

- source vertex x and sink v
- for a path P_1 arriving at v let (a, s), where
  - a : time of arrival at v
  - s : time of departure from x
- consider another path P_2 arriving at v with (a', s')

- if $(s' > s \text{ and } a' \leq a)$ or $(s' = s \text{ and } a' < a)$
  - then path P_2 dominates path P_1
  - because $a' - s' < a - s$

Dominating Path

(Wu et al., 2014)

- source vertex x and sink v
- for a path P_1 arriving at v let (a, s), where
  - a : time of arrival at v
  - s : time of departure from x
- consider another path P_2 arriving at v with (a', s')
- if $(s' > s \text{ and } a' \leq a)$ or $(s' = s \text{ and } a' < a)$
  - then path P_2 dominates path P_1
  - because $a' - s' < a - s$
- we can replace P_1 with P_2 and improve duration

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node
  - when new edge arrives extend non-dominated path

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node
  - when new edge arrives extend non-dominated path

Shortest and Minimum Hop Path

- similar to algorithm for fastest path

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node
  - when new edge arrives extend non-dominated path

Shortest and Minimum Hop Path

- similar to algorithm for fastest path
  - but keep track of non-dominated path wrt to the
  - transition times, or

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node
  - when new edge arrives extend non-dominated path

Shortest and Minimum Hop Path

- similar to algorithm for fastest path
  - but keep track of non-dominated path wrt to the
  - transition times, or
  - number of interactions, instead of the duration

Finding Optimal Path

Fastest Path (Wu et al., 2014)

- streaming algorithm can be adapted using dominating paths
  - keep a list of non-dominated labels at each node
  - when new edge arrives extend non-dominated path

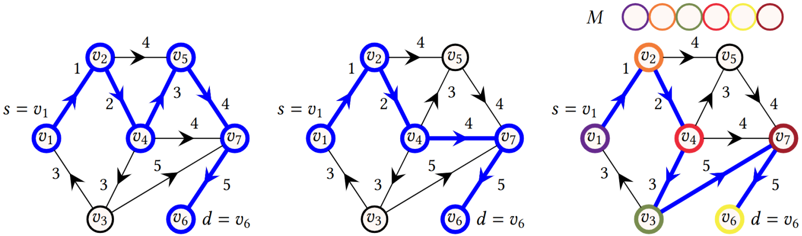
Shortest and Minimum Hop Path

- similar to algorithm for fastest path
  - but keep track of non-dominated path wrt to the
  - transition times, or
  - number of interactions, instead of the duration

Linear time algorithms in case of equal transition times

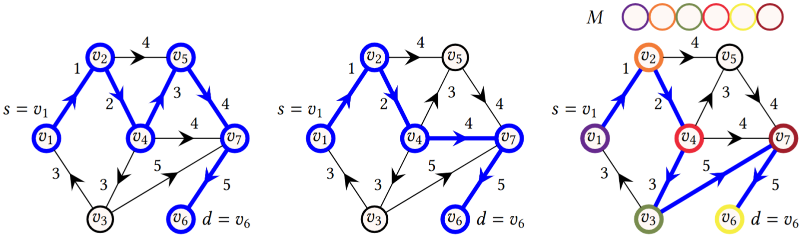
Restless Walks and Paths

- function $\beta : V \to \mathbb{R}$ determines the maximum waiting time at each node



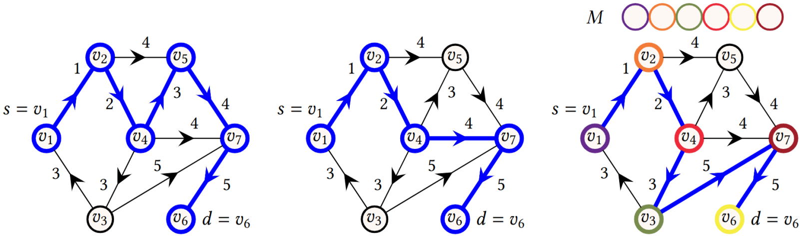
Restless Walks and Paths

- function $\beta : V \to \mathbb{R}$ determines the maximum waiting time at each node
  - motivation, e.g., spreading of disease



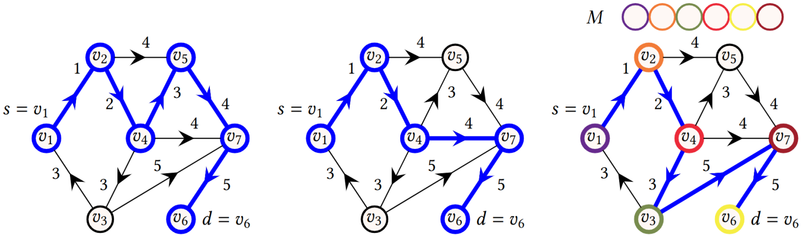
Restless Walks and Paths

- function $\beta : V \to \mathbb{R}$ determines the maximum waiting time at each node
  - motivation, e.g., spreading of disease
- finding restless temporal paths is NP-hard (Casteigts et al., 2021)



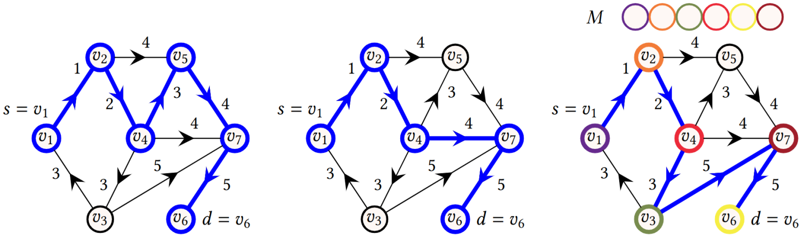
Restless Walks and Paths

- function $\beta : V \to \mathbb{R}$ determines the maximum waiting time at each node
  - motivation, e.g., spreading of disease
- finding restless temporal paths is NP-hard (Casteigts et al., 2021)
- finding restless temporal walks possible in $\mathcal{O}(|V| + |E| \log |E|)$ (Himmel et al., 2019)



Restless Walks and Paths

- function $\beta : V \to \mathbb{R}$ determines the maximum waiting time at each node
  - motivation, e.g., spreading of disease
- finding restless temporal paths is NP-hard (Casteigts et al., 2021)
- finding restless temporal walks possible in $\mathcal{O}(|V| + |E| \log |E|)$ (Himmel et al., 2019)
- extended for colored restless path and reachability (Thejaswi et al., 2020)



Static Expansion of a Temporal Network

- transformation of a temporal network to a directed static network
  - temporal paths in temporal network correspond to static paths in the directed static network
- how to create such a transformation?

Static Expansion of a Temporal Network

- transformation of a temporal network to a directed static network
  - temporal paths in temporal network correspond to static paths in the directed static network

- how to create such a transformation?

1. create a copy of each node for each time instance

Static Expansion of a Temporal Network

- transformation of a temporal network to a directed static network
  - temporal paths in temporal network correspond to static paths in the directed static network

- how to create such a transformation?

1. create a copy of each node for each time instance
2. create a directed edge from the $(t-1)$-th copy of u to the t-th copy of u, for all nodes u, and all time instances t

Static Expansion of a Temporal Network

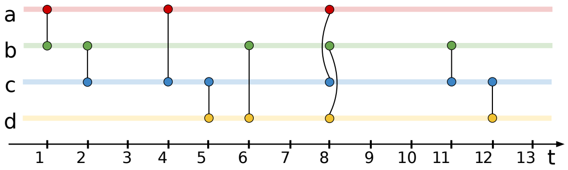
- transformation of a temporal network to a directed static network
  - temporal paths in temporal network correspond to static paths in the directed static network

- how to create such a transformation?

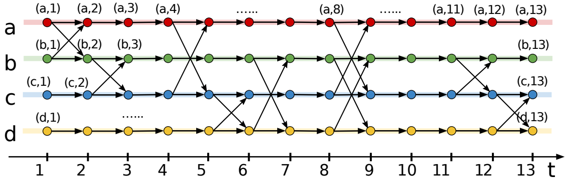
1. create a copy of each node for each time instance
2. create a directed edge from the $(t-1)$-th copy of u to the t-th copy of u, for all nodes u, and all time instances t
3. create directed edges for the temporal edges

Static Expansion Graphs

- $V_e = \{(v, t) \mid v \in V,\ t \in T\}$, where T is the set of all possible timestamps
- edges E_e : interactions between the temporal nodes V_t

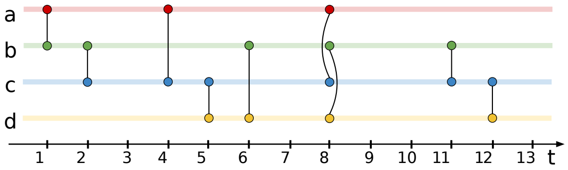


temporal graph with transition time 1

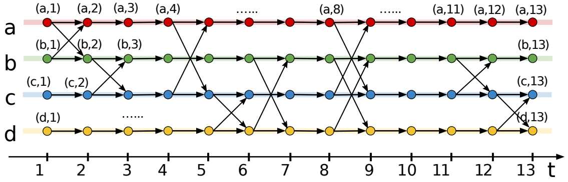


Static Expansion Graphs

- $V_e = \{(v,t) \mid v \in V,\ t \in T\}$, where T is the set of all possible timestamps
- edges E_e : interactions between the temporal nodes V_t



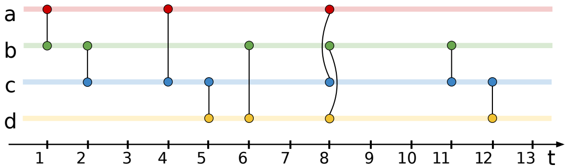
temporal graph with transition time 1



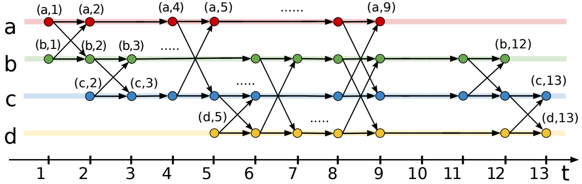
Problem: Size in number of time-steps $\Theta(|T| \cdot |V| + |E|)$

Static Expansion Graphs

- $V_e = \{(v,t) \mid v \in V,\ t \in T(v)\}$, where $T(v)$ is the set of times with activity at v
- edges E_e : interactions between the temporal nodes V_t

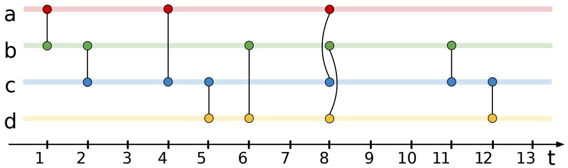


temporal graph with transition time 1

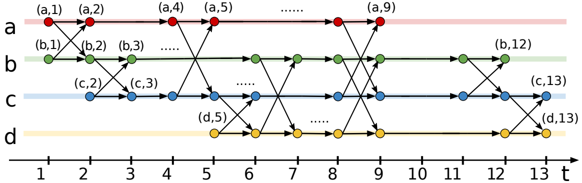


Static Expansion Graphs

- $V_e = \{(v,t) \mid v \in V,\ t \in T(v)\}$, where $T(v)$ is the set of times with activity at v
- edges E_e : interactions between the temporal nodes V_t



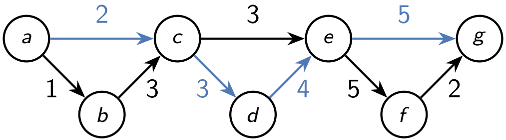
temporal graph with transition time 1



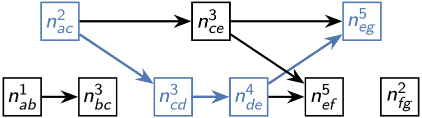
Size in number of edges $\mathcal{O}(|E|)$

Directed Line Graph

Temporal graph



Directed line graph



- temporal walk in G of length $\ell + 1 \Leftrightarrow$ walk of length ℓ in $D(G)$
- counting walks by matrix powers of adjacency matrix
- size in $\mathcal{O}(|E|^2)$

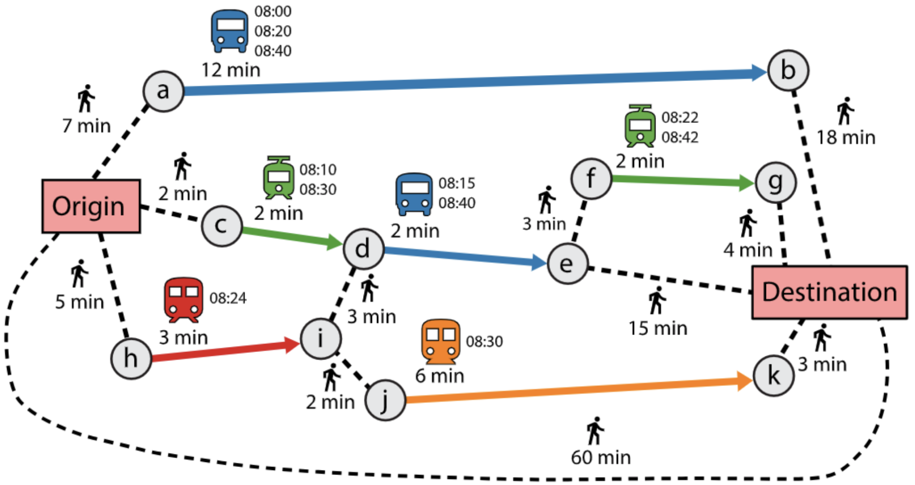
Static Representations

- static expansion graph and directed line graph are directed acyclic graphs
  if edges have non-zero transition times

- standard graph algorithms (bfs, dfs, Dijkstra, Bellman-Ford) can be adopted for finding
  - optimal temporal paths and temporal walks

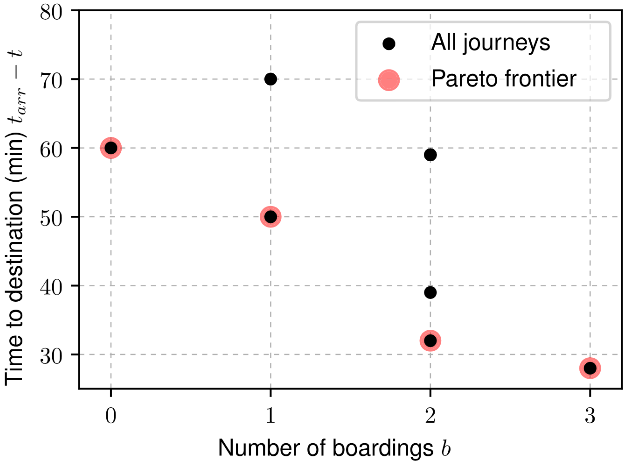
- upstream, downstream reachability sets

Transportation Temporal Networks



(Kujala et al., 2018)

Pareto-optimal Journeys

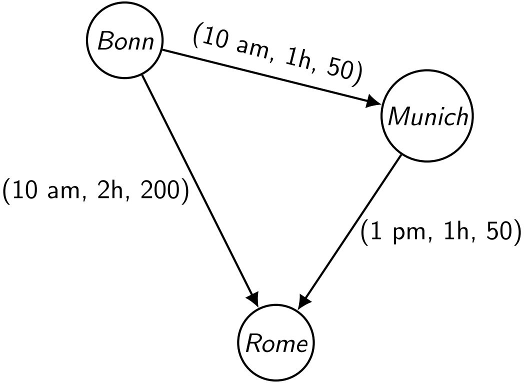


(Kujala et al., 2018)

Pareto-optimal Journeys

Weighted Temporal Graph

- Additional edge costs (u, v, t, λ, c) with $c \in \mathbb{R}$



Bicriteria optimal paths

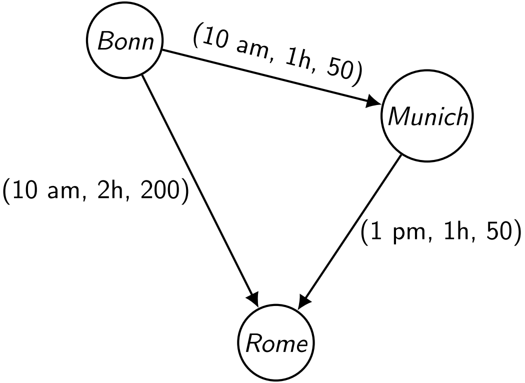
- solution: pair (duration, costs)

- non-stop: fast but expensive (2h, 200)
- via Munich: slow but cheap (4h, 100)

Pareto-optimal Journeys

Weighted Temporal Graph

- Additional edge costs (u, v, t, λ, c) with $c \in \mathbb{R}$



Bicriteria optimal paths

- solution: pair (duration, costs)
- non-stop: fast but expensive (2h, 200)
- via Munich: slow but cheap (4h, 100)

Enumeration of temporal paths that are efficient wrt. duration and cost in polynomial delay and space

(Mutzel and Oettershagen, 2019)

Temporal Graph Properties

- many static graph properties need to be adapted for temporal graphs to be meaningful
- local and global properties, often several variants with different focus

Temporal Graph Properties

- many static graph properties need to be adapted for temporal graphs to be meaningful
- local and global properties, often several variants with different focus

- diameter
  - shortest latency of time-respecting paths over connected pairs (Chaintreau et al., 2007)
  - restricted on connected pairs, as real data have many disconnected pairs

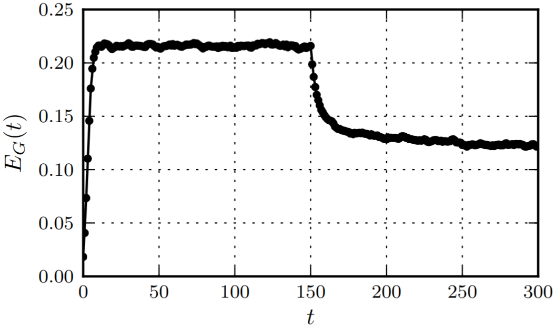
  - the minimum integer d for which the duration between each pair of nodes $u, v \in V$ is at most d (over all possible starting times) (Michail, 2016)

Temporal Network Efficiency

- network efficiency : the harmonic mean of durations (latency) over all pairs (Tang et al., 2009)

$$E(t_1, t_2) = \frac{1}{n(n-1)} \sum_{u,v \in V} \frac{1}{d_{(t_1,t_2)}(u,v)}$$

- application: robustness of network (Scellato et al., 2011)



drop in efficiency: at time $t = 150$, 20% of nodes are removed (sliding time window)

Burstiness

(Goh and Barabási, 2008)

- defined for sequence of inter-event times τ (of single node or pair of nodes, or global)
- measures deviation from memoryless random Poisson process
- defined as

$$B(\tau) = \frac{\sigma_\tau - m_\tau}{\sigma_\tau + m_\tau} \in [-1, 1],$$

where σ_τ and m_τ denote the standard deviation and mean of the inter-contact times τ

Burstiness

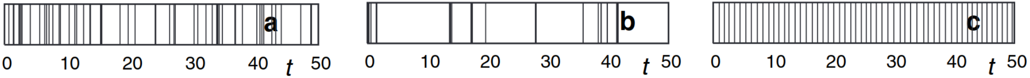
(Goh and Barabási, 2008)

- defined for sequence of inter-event times τ (of single node or pair of nodes, or global)
- measures deviation from memoryless random Poisson process
- defined as

$$B(\tau) = \frac{\sigma_\tau - m_\tau}{\sigma_\tau + m_\tau} \in [-1, 1],$$

where σ_τ and m_τ denote the standard deviation and mean of the inter-contact times τ

- (a) $B(\tau) = 0$ $\Rightarrow$ a Poisson distribution
  (b) $B(\tau) = 1$ $\Rightarrow$ a maximally bursty sequence
  (c) $B(\tau) = -1$ $\Rightarrow$ a periodic sequence



Topological Overlap

- quantifies the persistency of edges through time (Tang et al., 2010b)
- the topological overlap is defined as

$$T_{\mathrm{to}}(G) = \frac{1}{n} \sum_{u \in V} \frac{1}{T} \sum_{t=1}^{T-1} \frac{\sum_{v \in N(u)} \phi_{uv}^{t} \phi_{uv}^{t+1}}{\sqrt{\sum_{v \in N(u)} \phi_{uv}^{t} \sum_{v \in N(u)} \phi_{uv}^{t+1}}} \in [0, 1],$$

where $\phi_{uv}^{t} = 1$ iff. there exists a temporal edges between u and v at time t and zero otherwise

Topological Overlap

- quantifies the persistency of edges through time (Tang et al., 2010b)
- the topological overlap is defined as

$$T_{\text{to}}(G) = \frac{1}{n} \sum_{u \in V} \frac{1}{T} \sum_{t=1}^{T-1} \frac{\sum_{v \in N(u)} \phi_{uv}^{t} \phi_{uv}^{t+1}}{\sqrt{\sum_{v \in N(u)} \phi_{uv}^{t} \sum_{v \in N(u)} \phi_{uv}^{t+1}}} \in [0, 1],$$

  where $\phi_{uv}^{t} = 1$ iff. there exists a temporal edges between u and v at time t and zero otherwise

- value close to zero: many edges change between consecutive time steps

Topological Overlap

- quantifies the persistency of edges through time (Tang et al., 2010b)
- the topological overlap is defined as

$$T_{\text{to}}(G) = \frac{1}{n} \sum_{u \in V} \frac{1}{T} \sum_{t=1}^{T-1} \frac{\sum_{v \in N(u)} \phi_{uv}^{t} \phi_{uv}^{t+1}}{\sqrt{\sum_{v \in N(u)} \phi_{uv}^{t} \sum_{v \in N(u)} \phi_{uv}^{t+1}}} \in [0, 1],$$

  where $\phi_{uv}^{t} = 1$ iff. there exists a temporal edges between u and v at time t and zero otherwise

- value close to zero: many edges change between consecutive time steps
- value close to one: means there are often only a few changes.

Temporal Clustering Coefficient

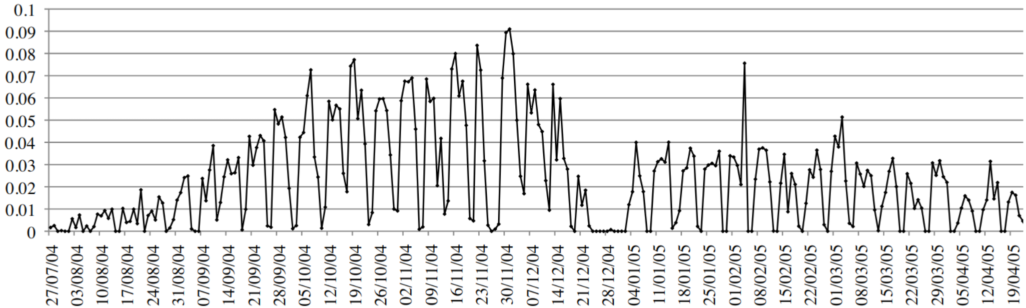
- the temporal clustering coefficient of node u in time interval I is defined as (Tang et al., 2009)

$$C_C(u, I) = \frac{\sum_{t \in I} \pi_t(u)}{|I| \binom{|N(u)|}{2}},$$

  where $\pi_t(u) =$ number of edges between neighbors of u at time t

- adaption of static clustering coefficient
- quantifies how close a nodes neighbors are to being a clique during time interval I

Temporal Clustering Coefficient



- human contact network at MIT campus using bluetooth scanning every 5 minutes
- global temporal clustering coefficient for each day
- higher during middle of the week and no clustering on holidays

(Tang et al., 2009)

Centrality Measures – Finding Important Nodes

Centrality Measures

Task

- assign to each node $v \in V$ a centrality value $c(v)$
- the higher $c(v)$ the more important is v

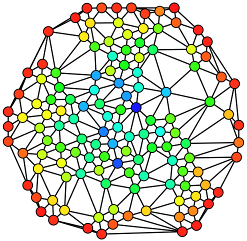
- many centrality measures on static graphs:
  e.g., degree, closeness, betweenness, Katz centrality, PageRank, ...

Centrality Measures

Task

- assign to each node $v \in V$ a centrality value $c(v)$
- the higher $c(v)$ the more important is v

- many centrality measures on static graphs:
  e.g., degree, closeness, betweenness, Katz centrality, PageRank, ...



Many important applications:

- identify key players, super spreaders, important persons, ...
- ranking web pages
- H-index used for ranking academics

Temporal Centrality Measures

- many common centrality measures are walk or path based
- classification in medial and radial (Borgatti and Everett, 2006)

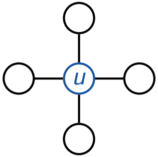
Temporal Centrality Measures

- many common centrality measures are walk or path based
- classification in medial and radial

(Borgatti and Everett, 2006)

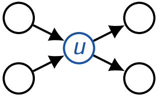
radial: captures node influence over its neighbors

- count incoming or outgoing walks or paths



medial: captures node role as intermediary

- count walks or paths passing node

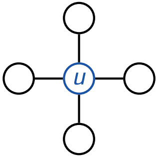


Temporal Centrality Measures

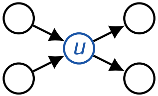
- many common centrality measures are walk or path based
- classification in medial and radial

(Borgatti and Everett, 2006)

radial: captures node influence over its neighbors
- count incoming or outgoing walks or paths



medial: captures node role as intermediary
- count walks or paths passing node



Common approach for temporal networks:
- replace path or walks with time-respecting paths or walks

Temporal Centrality

Name	Type		Use-case
temporal degree	radial	-	identify nodes with high degree
temporal closeness	radial	paths	identify nodes that can reach other nodes fast (or can be reached fast)
temporal pagerank	radial	walks	adapts static pagerank to capture concept drift
temporal katz	radial	walks	identify nodes with many incoming temporal walks
temporal H-index	radial	walks	identification of super-spreaders
temporal betweenness	medial	paths	identify nodes passed by many optimal temporal paths
temporal walk centrality	medial	walks	identify nodes that can obtain and distribute information

- choosing the right centrality measure depends on use-case
- many further temporal centrality variants, e.g., temporal eigenvector, temporal gravity, etc. (Hu et al., 2015; Rocha and Masuda, 2014; Tang et al., 2010a; Tsalouchidou et al., 2020; Bi et al., 2021; Elmezain et al., 2021; Zaoli et al., 2019; Tao et al., 2022; Taylor et al., 2021; Rozenshtein and Gionis, 2016) ...

Temporal Centrality

Name	Type		Use-case
temporal degree	radial	-	identify nodes with high degree
temporal closeness	radial	paths	identify nodes that can reach other nodes fast (or can be reached fast)
temporal pagerank	radial	walks	adapts static pagerank to capture concept drift
temporal katz	radial	walks	identify nodes with many incoming temporal walks
temporal H-index	radial	walks	identification of super-spreaders
temporal betweenness	medial	paths	identify nodes passed by many optimal temporal paths
temporal walk centrality	medial	walks	identify nodes that can obtain and distribute information

- choosing the right centrality measure depends on use-case
- many further temporal centrality variants, e.g., temporal eigenvector, temporal gravity, etc. (Hu et al., 2015; Rocha and Masuda, 2014; Tang et al., 2010a; Tsalouchidou et al., 2020; Bi et al., 2021; Elmezain et al., 2021; Zaoli et al., 2019; Tao et al., 2022; Taylor et al., 2021; Rozenshtein and Gionis, 2016) ...

From Static to Temporal Closeness

Static harmonic closeness:

$$C_c(u) = \sum_{v \in V \setminus \{u\}} \frac{1}{d_s(u, v)}$$

- $d_s(u, v)$ is shortest path distance
- high centrality means short paths to many other nodes
- temporal: replace $d_s(u, v)$ with temporal distance
- several different variants
  (Wu et al., 2014; Crescenzi et al., 2020; Tang et al., 2010a; Santoro et al., 2011; Gao et al., 2015)

Temporal Closeness

Harmonic temporal closeness for $u \in V$:

$$c(u) = \sum_{v \in V \setminus \{u\}} \frac{1}{d(u, v)}$$

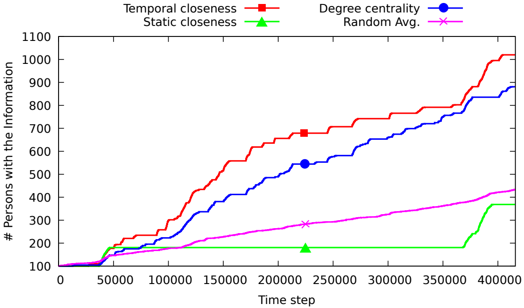
$d(u, v)$ is the minimum duration distance (i.e., arrival time - starting time).

Use case

- find nodes that spread information fast

Computation:

- call minimum duration streaming algorithm (Wu et al., 2014) for each node
- lack of scalability



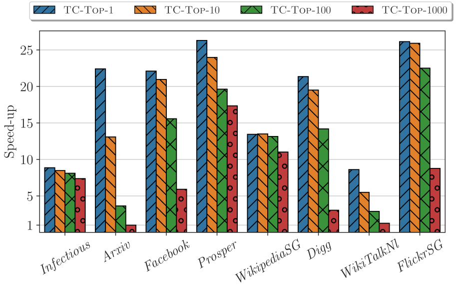
(Oettershagen and Mutzel, 2022)

Temporal Closeness – Top-k Computation

Top-k closeness problem: find all nodes with one of the k topmost closeness values

Top-k closeness computation

- for each vertex $u \in V$
  - run min. duration algorithm to compute $d(u, v)$ for all $v \in V$
    - if upper bound of $c(u)$ is smaller than k-th largest value: stop computation early



(Oettershagen and Mutzel, 2022)

Temporal Closeness – Index

Problem: rank all nodes according to temporal closeness.

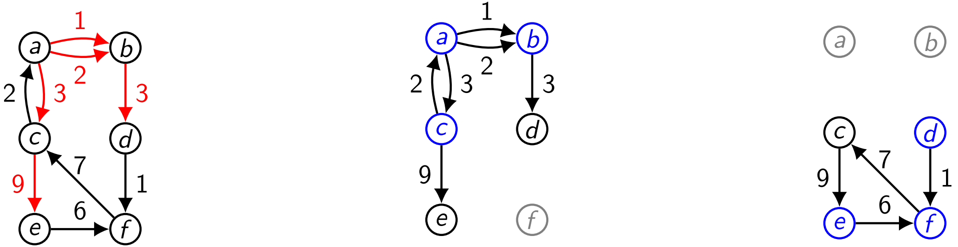
Indexing approach

- index to speed up minimum duration computation
- two phases: (i) indexing and (ii) query phase

(Oettershagen and Mutzel, 2023)

Temporal Closeness – Index

- **Construction:**
  - construct k subgraphs $\{S_1, \ldots, S_k\} = \mathcal{S}$
  - find mapping $f : V \to \mathcal{S}$ that assigns to each $S_j \in \mathcal{S}$ all vertices $v \in V$ s.t. all edges reachable from v are in $f(v) = S_j$
  - minimize size $\max_{S \in \mathcal{S}}\{|S|\}$



(a) G (b) S_1 (c) S_2

Optimal assignment is NP-hard

Temporal Closeness – Index

Approximation ratio of greedy:

$$\frac{\mathit{size}(\textsc{Greedy})}{\mathit{size}(\textsc{Opt})} \leq \frac{k}{\delta},$$

with $1 \leq \delta \leq k$ depending on the topology of the graph

Time complexity: $\mathcal{O}(nmk)$

- $n = |V|$ rounds, $m = |E|$
- each round determine S_j such that greedy choice is minimal in $\mathcal{O}(m)$ for $j \in \{1, \ldots, k\}$

Shared memory parallelization: $\mathcal{O}(\frac{nmk}{P})$ using P processors (CREW)

Temporal Closeness – Index

OOT—Out of time after 7 days. number of subgraphs $k = 2048$.

Data set	$\|V\|$	$\|E\|$	BASELINE	TOP-100	SUBSTREAM
Infectious	10 972	415 912	12.06 s	2.25 s	**1.51** s
AskUbuntu	159 316	964 437	229.73 s	132.53 s	**102.46** s
Prosper	89 269	3 394 978	1 665.20 s	260.87 s	**109.33** s
Arxiv	28 093	4 596 803	630.60 s	398.50 s	**286.86** s
Youtube	3 223 585	9 375 374	145.98 h	81.21 h	**59.72** h
StackOverflow	2 464 606	17 823 525	OOT	107.66 h	**86.49** h

- BASELINE: Streaming algorithm (Wu et al., 2014)
- TOP-100: Top-k algorithm with $k = 100$
- SUBSTREAM: Index based computation

Temporal Centrality

Name	Type		Use-case
temporal degree	radial	-	identify nodes with high degree
temporal closeness	radial	paths	identify nodes that can reach other nodes fast (or can be reached fast)
temporal pagerank	radial	walks	adapts static pagerank to capture concept drift
temporal katz	radial	walks	identify nodes with many incoming temporal walks
temporal H-index	radial	walks	identification of super-spreaders
temporal betweenness	medial	paths	identify nodes passed by many optimal temporal paths
temporal walk centrality	medial	walks	identify nodes that can obtain and distribute information

- choosing the right centrality measure depends on use-case
- many further temporal centrality variants, e.g., temporal eigenvector, temporal gravity, etc. (Hu et al., 2015; Rocha and Masuda, 2014; Tang et al., 2010a; Tsalouchidou et al., 2020; Bi et al., 2021; Elmezain et al., 2021; Zaoli et al., 2019; Tao et al., 2022; Taylor et al., 2021; Rozenshtein and Gionis, 2016) ...

Static H-Index

- the H-index was originally proposed by J. E. Hirsch 2005
  $\rightarrow$ measuring the productivity and impact of scientists
- the maximum value of h such that the author has published at least h papers that have each been cited at least h times

paper ID	citations	counted in H-index
1	25	Yes
2	18	Yes
3	12	Yes
4	9	Yes
5	7	Yes
6	5	No
7	3	No

Static H-Index

- the H-index was originally proposed by J. E. Hirsch 2005
  → measuring the productivity and impact of scientists
- the maximum value of h such that the author has published at least h papers that have each been cited at least h times

paper ID	citations	counted in H-index
1	25	Yes
2	18	Yes
3	12	Yes
4	9	Yes
5	7	Yes
6	5	No
7	3	No

Recently used for quantifying spreading influence (Lü et al., 2016)

Static H-Index

- $\mathcal{H}\colon \mathcal{M} \to \mathbb{N}_0$ returns for a multiset of integers $S \subseteq \{\!\{s \mid s \in \mathbb{N}_0\}\!\}$ the maximum integer i such that there are at least i elements s in S with $s \geq i$

Static H-Index

- $\mathcal{H}\colon \mathcal{M} \to \mathbb{N}_0$ returns for a multiset of integers $S \subseteq \{\!\{s \mid s \in \mathbb{N}_0\}\!\}$ the maximum integer i such that there are at least i elements s in S with $s \geq i$

n-th order H-index $s_u^{(n)}$ of a node u in a static graph:

- let $s_u^{(0)} = \delta(u)$ the degree of node u, then

$$s_u^{(n)} = \mathcal{H}\left(\{\!\{s_v^{(n-1)} \mid v \in V \text{ and } v \text{ is neighbor of } u\}\!\}\right)$$

- the value of $s_u^{(1)}$ corresponds to the H-index of u

(Lü et al., 2016)

n-th Order Temporal H-Index

- the multiset $\mathcal{N}(v, t)$ contains all pairs of nodes and times (w, t_w) such that there is a temporal edge from v to w leaving at time $t' \geq t$ and arriving at time t_w

n-th Order Temporal H-Index

- the multiset $\mathcal{N}(v,t)$ contains all pairs of nodes and times (w,t_w) such that there is a temporal edge from v to w leaving at time $t' \geq t$ and arriving at time t_w

Definition

The *n*-th order temporal H-index of a node $v \in V$ is defined as $h_v^{(n)} = h_{v,0}^{(n)}$ with

$$h_{v,t}^{(n)} = \mathcal{H}\left(\left\{\!\!\left\{ h_{w,t_w}^{(n-1)} \;\middle|\; (w,t_w) \in \mathcal{N}(v,t) \right\}\!\!\right\}\right),$$

and $h_{v,t}^{(0)} = |\mathcal{N}(v,t)|$.

n-th Order Temporal H-Index

- the multiset $\mathcal{N}(v,t)$ contains all pairs of nodes and times (w, t_w) such that there is a temporal edge from v to w leaving at time $t' \geq t$ and arriving at time t_w

Definition

The *n*-th order temporal H-index of a node $v \in V$ is defined as $h_v^{(n)} = h_{v,0}^{(n)}$ with

$$h_{v,t}^{(n)} = \mathcal{H}\left(\left\{\!\!\left\{ h_{w,t_w}^{(n-1)} \;\middle|\; (w, t_w) \in \mathcal{N}(v,t) \right\}\!\!\right\}\right),$$

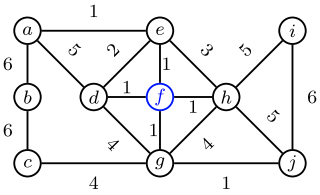
and $h_{v,t}^{(0)} = |\mathcal{N}(v,t)|$.

Computation:

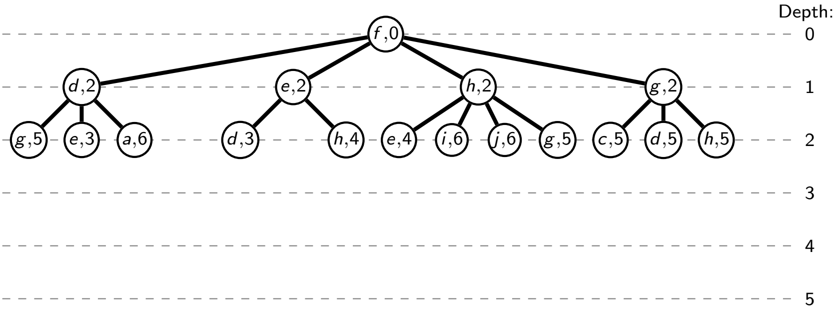
- single-pass streaming algorithm for each node i-th order H-indices for $0 \leq i \leq n$
- running time in $\mathcal{O}(|E| n \delta_{\max})$ and space in $\mathcal{O}(|V| n \delta_{\max})$

(Oettershagen et al., 2023b)

n-th Order Temporal H-Index



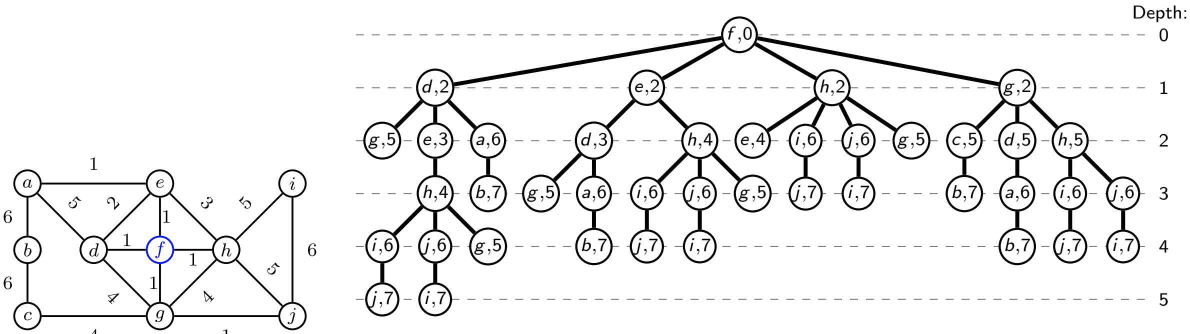
(a) Temporal network $\mathcal{G}$.



(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

$$h^{(1)}_{f,0} = \mathcal{H}(\{\!\{h^{(0)}_{d,2}, h^{(0)}_{e,2}, h^{(0)}_{h,2}, h^{(0)}_{g,2}\}\!\}) = \mathcal{H}(\{\!\{3, 2, 4, 3\}\!\}) = 3$$

n-th Order Temporal H-Index

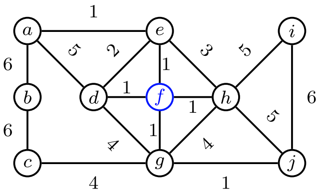


(a) Temporal network $\mathcal{G}$.

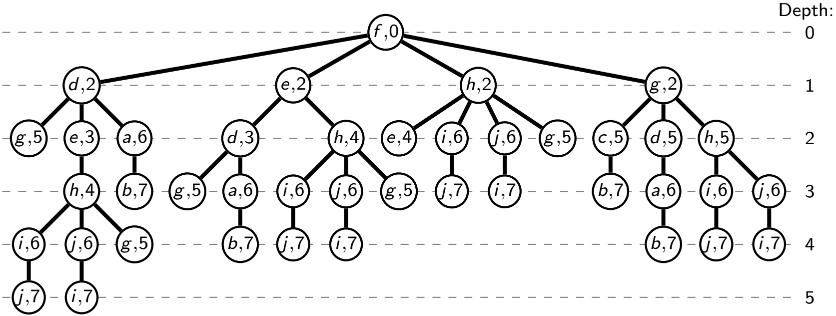
(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

$$h^{(2)}_{f,0} =$$

n-th Order Temporal H-Index



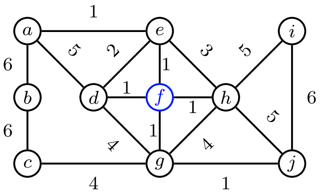
(a) Temporal network $\mathcal{G}$.



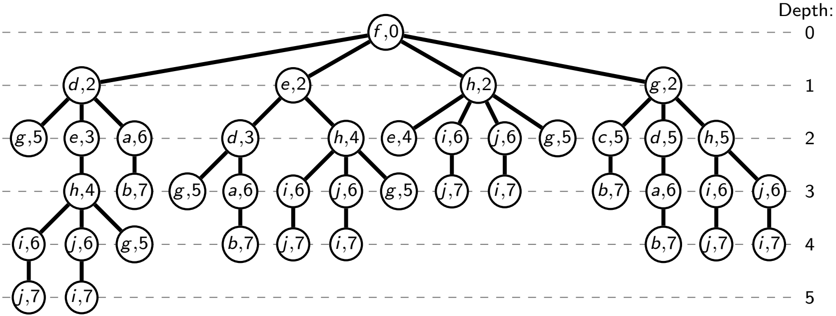
(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

$$h^{(2)}_{f,0} = \mathcal{H}(\{\!\{h^{(1)}_{d,2}, h^{(1)}_{e,2}, h^{(1)}_{h,2}, h^{(1)}_{g,2}\}\!\})$$

n-th Order Temporal H-Index



(a) Temporal network $\mathcal{G}$.

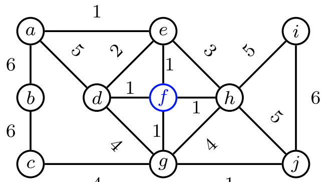


(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

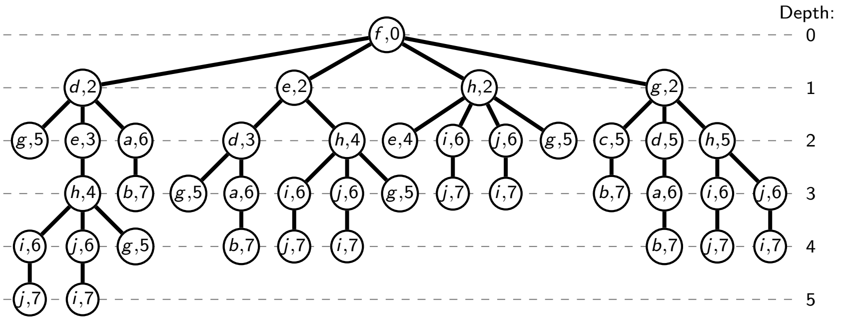
$$h^{(2)}_{f,0} = \mathcal{H}(\{\!\{ h^{(1)}_{d,2}, h^{(1)}_{e,2}, h^{(1)}_{h,2}, h^{(1)}_{g,2} \}\!\})$$

$$= \mathcal{H}(\{\!\{ \mathcal{H}(\{\!\{ h^{(0)}_{g,5}, h^{(0)}_{e,3}, h^{(0)}_{a,6} \}\!\}), \mathcal{H}(\{\!\{ h^{(0)}_{d,3}, h^{(0)}_{h,4} \}\!\}), \mathcal{H}(\{\!\{ h^{(0)}_{e,4}, h^{(0)}_{i,6}, h^{(0)}_{j,6}, h^{(0)}_{g,5} \}\!\}), \mathcal{H}(\{\!\{ h^{(0)}_{c,5}, h^{(0)}_{d,5}, h^{(0)}_{h,5} \}\!\}) \}\!\}$$

n-th Order Temporal H-Index



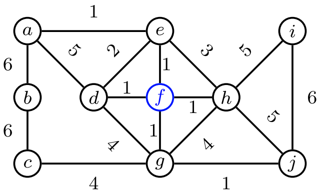
(a) Temporal network $\mathcal{G}$.



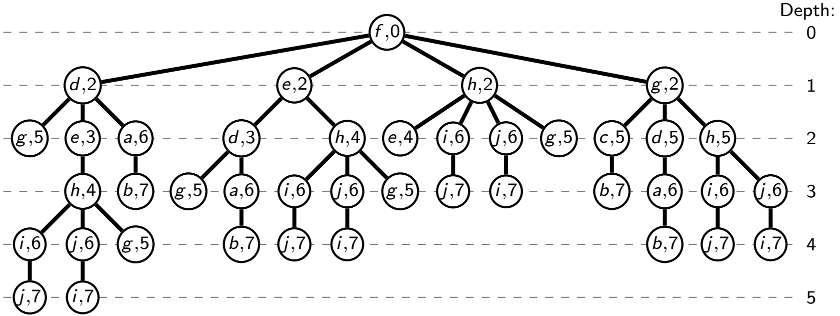
(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

$$
\begin{aligned}
h^{(2)}_{f,0} &= \mathcal{H}(\{\!\{h^{(1)}_{d,2}, h^{(1)}_{e,2}, h^{(1)}_{h,2}, h^{(1)}_{g,2}\}\!\}) \\
&= \mathcal{H}(\{\!\{\mathcal{H}(\{\!\{h^{(0)}_{g,5}, h^{(0)}_{e,3}, h^{(0)}_{a,6}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{d,3}, h^{(0)}_{h,4}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{e,4}, h^{(0)}_{i,6}, h^{(0)}_{j,6}, h^{(0)}_{g,5}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{c,5}, h^{(0)}_{d,5}, h^{(0)}_{h,5}\}\!\})\}\!\} \\
&= \mathcal{H}(\{\!\{\mathcal{H}(\{\!\{0,1,1\}\!\}), \mathcal{H}(\{\!\{2,3\}\!\}), \mathcal{H}(\{\!\{0,1,1,0\}\!\}), \mathcal{H}(\{\!\{1,1,2\}\!\})\}\!\})
\end{aligned}
$$

n-th Order Temporal H-Index



(a) Temporal network $\mathcal{G}$.



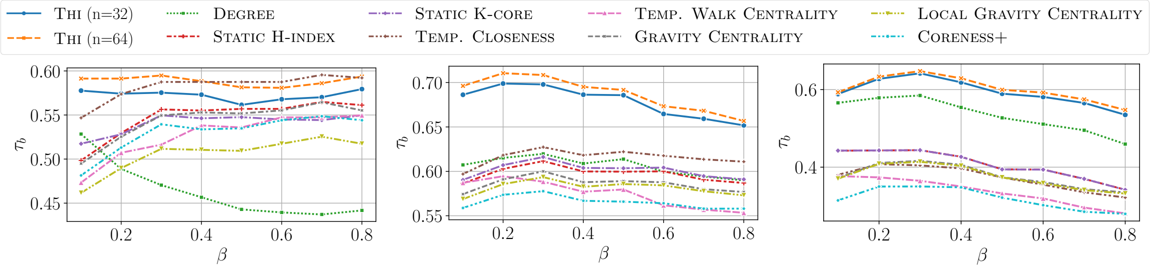
(b) The reachability tree $\Gamma(f)$ for vertex f in the temporal network shown in (a).

$$
\begin{aligned}
h^{(2)}_{f,0} &= \mathcal{H}(\{\!\{h^{(1)}_{d,2}, h^{(1)}_{e,2}, h^{(1)}_{h,2}, h^{(1)}_{g,2}\}\!\}) \\
&= \mathcal{H}(\{\!\{\mathcal{H}(\{\!\{h^{(0)}_{g,5}, h^{(0)}_{e,3}, h^{(0)}_{a,6}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{d,3}, h^{(0)}_{h,4}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{e,4}, h^{(0)}_{i,6}, h^{(0)}_{j,6}, h^{(0)}_{g,5}\}\!\}), \mathcal{H}(\{\!\{h^{(0)}_{c,5}, h^{(0)}_{d,5}, h^{(0)}_{h,5}\}\!\})\}\!\}) \\
&= \mathcal{H}(\{\!\{\mathcal{H}(\{\!\{0,1,1\}\!\}), \mathcal{H}(\{\!\{2,3\}\!\}), \mathcal{H}(\{\!\{0,1,1,0\}\!\}), \mathcal{H}(\{\!\{1,1,2\}\!\})\}\!\}) = \mathcal{H}(\{\!\{1,2,1,1\}\!\}) = 1
\end{aligned}
$$

n-th Order Temporal H-Index

Use Case: Influential spreader identification

- computed for different infection probabilities β the mean node influence R_u over 1000 independent SIR simulations leading to the SIR node rankings
- compared the SIR rankings with those obtained by the centrality measures using the Kendall τ_b rank correlation measure



(a) *Malawi* (b) *FacebookMsg* (c) *Email*

Temporal Centrality

Name	Type		Use-case
temporal degree	radial	-	identify nodes with high degree
temporal closeness	radial	paths	identify nodes that can reach other nodes fast (or can be reached fast)
temporal pagerank	radial	walks	adapts static pagerank to capture concept drift
temporal katz	radial	walks	identify nodes with many incoming temporal walks
temporal H-index	radial	walks	identification of super-spreaders
temporal betweenness	medial	paths	identify nodes passed by many optimal temporal paths
temporal walk centrality	medial	walks	identify nodes that can obtain and distribute information

- choosing the right centrality measure depends on use-case
- many further temporal centrality variants, e.g., temporal eigenvector, temporal gravity, etc. (Hu et al., 2015; Rocha and Masuda, 2014; Tang et al., 2010a; Tsalouchidou et al., 2020; Bi et al., 2021; Elmezain et al., 2021; Zaoli et al., 2019; Tao et al., 2022; Taylor et al., 2021; Rozenshtein and Gionis, 2016) ...

Betweenness Centrality

- node importance of $u \in V$ in terms of number of optimal paths visiting u

$$B(u) = \sum_{s \neq u \neq z \in V} \frac{\sigma_{s,z}(u)}{\sigma_{s,z}}$$

  - $\sigma_{s,z}$ number of shortest s, z-paths
  - $\sigma_{s,z}(u)$ number of shortest s, z-paths visiting u

Betweenness Centrality

- node importance of $u \in V$ in terms of number of optimal paths visiting u

$$B(u) = \sum_{s \neq u \neq z \in V} \frac{\sigma_{s,z}(u)}{\sigma_{s,z}}$$

  - $\sigma_{s,z}$ number of shortest s, z-paths
  - $\sigma_{s,z}(u)$ number of shortest s, z-paths visiting u

- Brandes' algorithm: iteratively calculates the shortest paths using dynamic programming, efficiently updating centrality scores

Betweenness Centrality

- node importance of $u \in V$ in terms of number of optimal paths visiting u

$$B(u) = \sum_{s \neq u \neq z \in V} \frac{\sigma_{s,z}(u)}{\sigma_{s,z}}$$

  - $\sigma_{s,z}$ number of shortest s, z-paths
  - $\sigma_{s,z}(u)$ number of shortest s, z-paths visiting u

- Brandes' algorithm: iteratively calculates the shortest paths using dynamic programming, efficiently updating centrality scores

- idea: replace shortest paths with optimal temporal paths (Kim and Anderson, 2012)

Betweenness Centrality

- node importance of $u \in V$ in terms of number of optimal paths visiting u

$$B(u) = \sum_{s \neq u \neq z \in V} \frac{\sigma_{s,z}(u)}{\sigma_{s,z}}$$

  - $\sigma_{s,z}$ number of shortest s, z-paths
  - $\sigma_{s,z}(u)$ number of shortest s, z-paths visiting u

- Brandes' algorithm: iteratively calculates the shortest paths using dynamic programming, efficiently updating centrality scores

- idea: replace shortest paths with optimal temporal paths (Kim and Anderson, 2012)

- computing betweenness values is at least as hard as counting optimal paths

Temporal Betweenness

Overview over the complexity of computing temporal betweenness centrality

path type	**strict**	**non-strict**
min.-hop	$\mathcal{O}(n^3 \cdot T^2)$	$\mathcal{O}(n^3 \cdot T^2)$
earliest arrival	#P-hard	#P-hard
fastest	#P-hard	#P-hard
prefix-earliest arrival	$\mathcal{O}(n \cdot m \cdot \log m)$	#P-hard
min.-hop earliest arrival	$\mathcal{O}(n^3 \cdot T^2)$	$\mathcal{O}(n^3 \cdot T^2)$

Prefix-earliest arrival: every prefix of the temporal path is an earliest arrival path

- computation using adapted Brandes' algorithm (Buß et al., 2020)

Temporal Betweenness

- more possible temporal walks and path types (Rymar et al., 2021)
  - characterization of properties such that path are efficient countable

Temporal Betweenness

- more possible temporal walks and path types (Rymar et al., 2021)
  - characterization of properties such that path are efficient countable
- approximation algorithms (Santoro and Sarpe, 2022; Cruciani, 2023)
  - sampling-based approximations for different kinds of temporal path types

Temporal Betweenness

- more possible temporal walks and path types (Rymar et al., 2021)
  - characterization of properties such that path are efficient countable
- approximation algorithms (Santoro and Sarpe, 2022; Cruciani, 2023)
  - sampling-based approximations for different kinds of temporal path types
- comparison of different proxies for temporal betweenness (Becker et al., 2023)
  - replacing global centrality with local pass-through-degree

Temporal Centrality

Name	Type		Use-case
temporal degree	radial	-	identify nodes with high degree
temporal closeness	radial	paths	identify nodes that can reach other nodes fast (or can be reached fast)
temporal pagerank	radial	walks	adapts static pagerank to capture concept drift
temporal katz	radial	walks	identify nodes with many incoming temporal walks
temporal H-index	radial	walks	identification of super-spreaders
temporal betweenness	medial	paths	identify nodes passed by many optimal temporal paths
temporal walk centrality	medial	walks	identify nodes that can obtain and distribute information

- choosing the right centrality measure depends on use-case
- many further temporal centrality variants, e.g., temporal eigenvector, temporal gravity, etc. (Hu et al., 2015; Rocha and Masuda, 2014; Tang et al., 2010a; Tsalouchidou et al., 2020; Bi et al., 2021; Elmezain et al., 2021; Zaoli et al., 2019; Tao et al., 2022; Taylor et al., 2021; Rozenshtein and Gionis, 2016) ...

Temporal Walk Centrality

- let $Y_{in}(u, t)$ and $Y_{out}(u, t)$ be the sets of incoming and outgoing temporal walks, resp., at node u and time t.

Temporal Walk Centrality

- let $Y_{in}(u,t)$ and $Y_{out}(u,t)$ be the sets of incoming and outgoing temporal walks, resp., at node u and time t.
- for weighting functions $\tau_{\Phi_{in}}$ and $\tau_{\Phi_{out}}$, define

$$\boldsymbol{W}_{in}(u,t) = \sum_{\omega \in Y_{in}(u,t)} \tau_{\Phi_{in}}(\omega) \quad \text{and} \quad \boldsymbol{W}_{out}(u,t) = \sum_{\omega \in Y_{out}(u,t)} \tau_{\Phi_{out}}(\omega)$$

Temporal Walk Centrality

- let $Y_{in}(u,t)$ and $Y_{out}(u,t)$ be the sets of incoming and outgoing temporal walks, resp., at node u and time t.
- for weighting functions $\tau_{\Phi_{in}}$ and $\tau_{\Phi_{out}}$, define

$$\boldsymbol{W}_{in}(u,t) = \sum_{\omega \in Y_{in}(u,t)} \tau_{\Phi_{in}}(\omega) \quad \text{and} \quad \boldsymbol{W}_{out}(u,t) = \sum_{\omega \in Y_{out}(u,t)} \tau_{\Phi_{out}}(\omega)$$

Temporal Walk Centrality

The temporal walk centrality of a vertex $u \in V$ is

$$C(u) = \sum_{t_1, t_2 \in T(\mathcal{G}), t_1 \leq t_2} \left(\boldsymbol{W}_{in}(u,t_1) \cdot \boldsymbol{W}_{out}(u,t_2) \cdot \Phi_m(t_1,t_2) \right).$$

Temporal Walk Centrality

- let $Y_{in}(u,t)$ and $Y_{out}(u,t)$ be the sets of incoming and outgoing temporal walks, resp., at node u and time t.
- for weighting functions $\tau_{\Phi_{in}}$ and $\tau_{\Phi_{out}}$, define

$$\boldsymbol{W}_{in}(u,t) = \sum_{\omega \in Y_{in}(u,t)} \tau_{\Phi_{in}}(\omega) \quad \text{and} \quad \boldsymbol{W}_{out}(u,t) = \sum_{\omega \in Y_{out}(u,t)} \tau_{\Phi_{out}}(\omega)$$

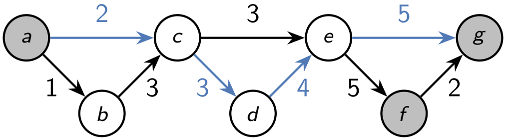
Temporal Walk Centrality

The temporal walk centrality of a vertex $u \in V$ is

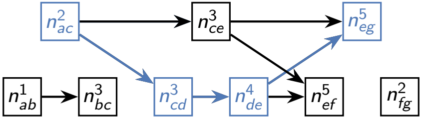
$$C(u) = \sum_{t_1, t_2 \in T(\mathcal{G}), t_1 \le t_2} \left(\boldsymbol{W}_{in}(u,t_1) \cdot \boldsymbol{W}_{out}(u,t_2) \cdot \Phi_m(t_1,t_2) \right).$$

Captures a nodes ability to obtain and pass on information

Temporal Walk Centrality

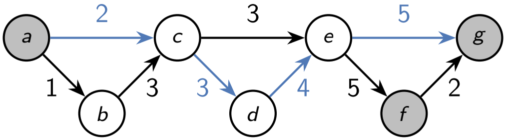


Directed line graph

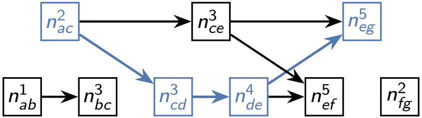


- temporal walk in G of length $\ell + 1 \Leftrightarrow$ walk of length ℓ in $D(G)$

Temporal Walk Centrality

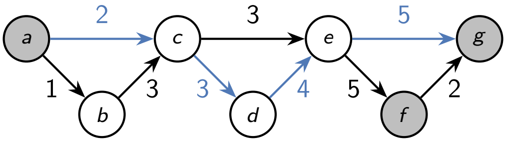


Directed line graph

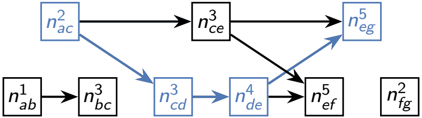


- temporal walk in G of length $\ell + 1 \Leftrightarrow$ walk of length ℓ in $D(G)$
- walks can be computed by matrix powers: Neumann series and the identity $\sum_{\ell=0}^{\infty} \boldsymbol{A}^{\ell} = (\boldsymbol{I} - \boldsymbol{A})^{-1}$ holds if the sum converges—guaranteed when largest absolute eigenvalue less than one

Temporal Walk Centrality



Directed line graph



- temporal walk in G of length $\ell + 1 \Leftrightarrow$ walk of length ℓ in $D(G)$
- walks can be computed by matrix powers: Neumann series and the identity $\sum_{\ell=0}^{\infty} \boldsymbol{A}^{\ell} = (\boldsymbol{I} - \boldsymbol{A})^{-1}$ holds if the sum converges—guaranteed when largest absolute eigenvalue less than one
- computation in $\mathcal{O}(|E|^{2.373})$ using matrix inversion
- approximation with power iteration in $\mathcal{O}(k(|E|^2))$

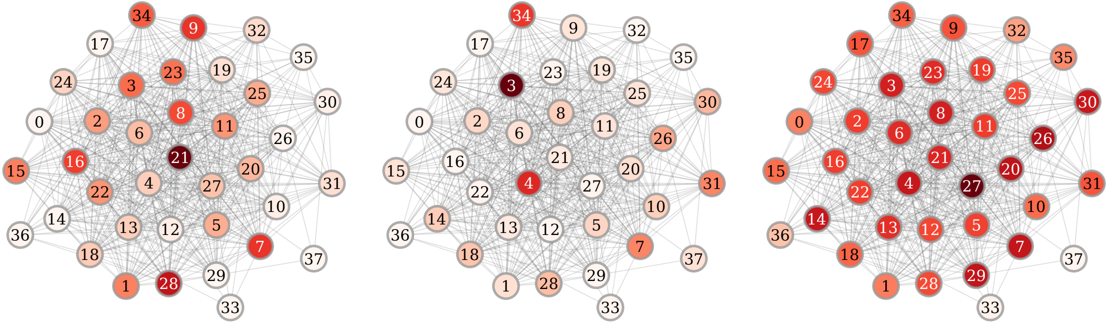
Temporal Walk Centrality

Two-Pass Streaming Algorithm

- **input:** edge sequence in chronological order, ties broken arbitrarily
- **forward pass** for computing incoming walks for $\boldsymbol{W}_{in}$
- **backward pass** for computing outgoing walks for $\boldsymbol{W}_{out}$

- running time $\mathcal{O}(|E| \cdot \tau_{max})$
- τ_{max} the largest cardinality of availability or arrival times at a node

Temporal Walk Centrality

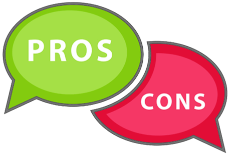


(a) temporal walk centr. (b) temporal between. (c) static walk between.

- *enron* email subgraph: of 38 nodes and 541 temporal edges.
- colors represent centrality value: darker $\rightarrow$ higher centrality.

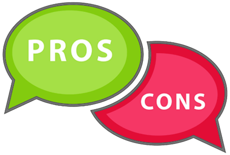
Temporal Centrality

- **Temporal closeness**
  - many different variants
  - intuitive, several approaches for improving computation times



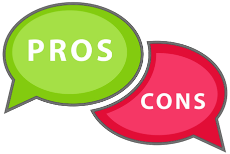
Temporal Centrality

- **Temporal closeness**
  - many different variants
  - intuitive, several approaches for improving computation times
- **Temporal H-index**
  - non-intuitive definition
  - able to capture the spreading capabilities well, efficient



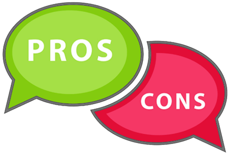
Temporal Centrality

- **Temporal closeness**
  - many different variants
  - intuitive, several approaches for improving computation times
- **Temporal H-index**
  - non-intuitive definition
  - able to capture the spreading capabilities well, efficient
- **Temporal betweenness**
  - not scalable and many variants hard to compute
  - intuitive, several approaches for improving computation times



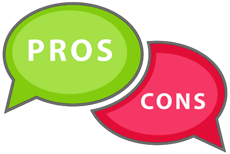
Temporal Centrality

- **Temporal closeness**
  - many different variants
  - intuitive, several approaches for improving computation times
- **Temporal H-index**
  - non-intuitive definition
  - able to capture the spreading capabilities well, efficient
- **Temporal betweenness**
  - not scalable and many variants hard to compute
  - intuitive, several approaches for improving computation times
- **Temporal walk centrality**
  - no ranking of sinks or sources
  - intuitive and efficient



Temporal Centrality

- **Temporal closeness**
  - many different variants
  - intuitive, several approaches for improving computation times
- **Temporal H-index**
  - non-intuitive definition
  - able to capture the spreading capabilities well, efficient
- **Temporal betweenness**
  - not scalable and many variants hard to compute
  - intuitive, several approaches for improving computation times
- **Temporal walk centrality**
  - no ranking of sinks or sources
  - intuitive and efficient

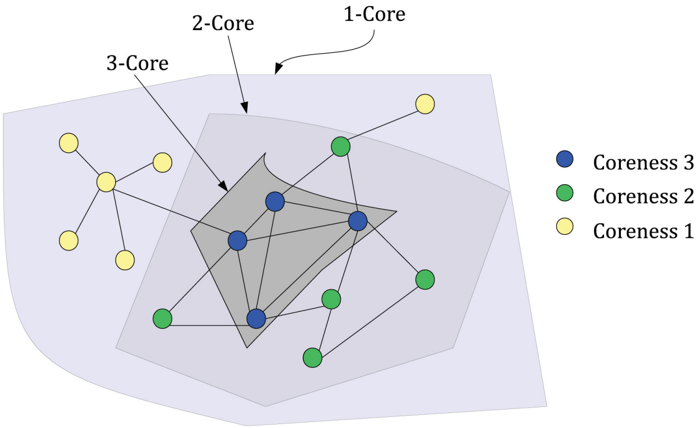


- choosing the right centrality measure depends on data and use-case

Temporal Core Decompositions

k-Core Decomposition

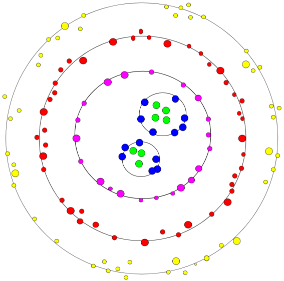
- k-core is a max. subgraph G_k of G, s.t. every node in G_k has at least k neighbors in G_k
- node u has core number $c(u) = k$ if u belongs to a k-core but not the $k+1$-core

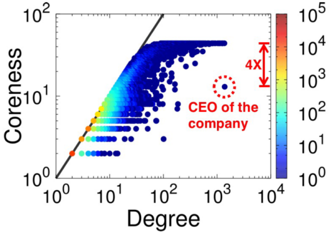


(Seidman, 1983; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks

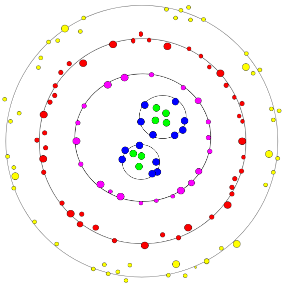


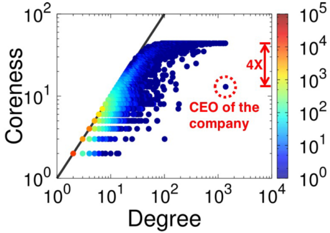


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection

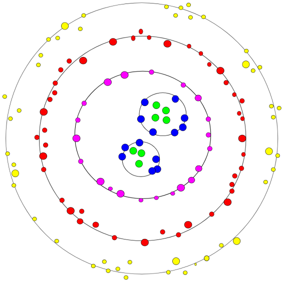


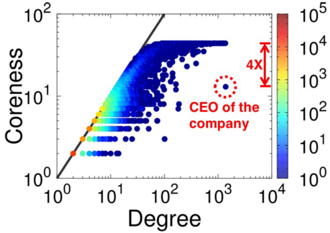


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection
- network visualization

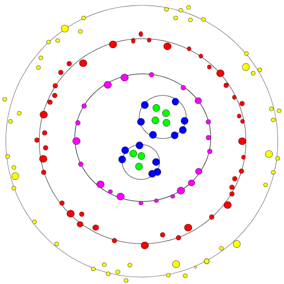


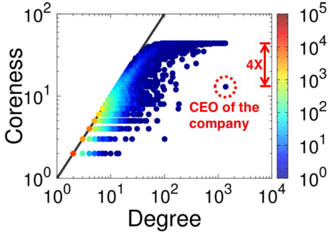


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection
- network visualization
- analyzing backbone structure of the internet

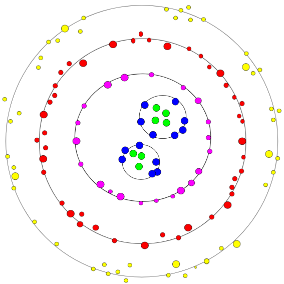


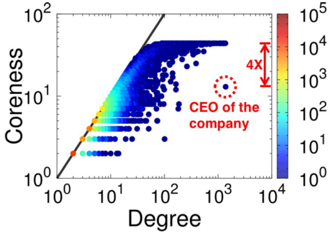


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection
- network visualization
- analyzing backbone structure of the internet
- network resilience and robustness

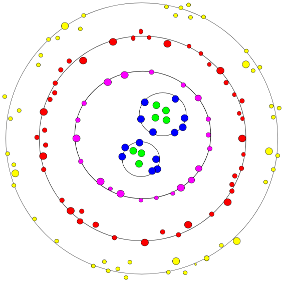


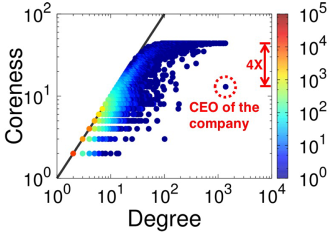


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection
- network visualization
- analyzing backbone structure of the internet
- network resilience and robustness
- study of spreading processes

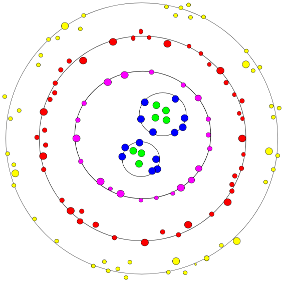


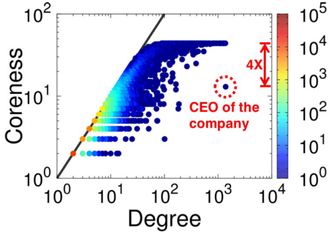


(Malliaros et al., 2020; Kong et al., 2019)

Applications

- identifying communities and dense graphs in social networks
- anomaly detection
- network visualization
- analyzing backbone structure of the internet
- network resilience and robustness
- study of spreading processes
- analyzing functional brain networks





(Malliaros et al., 2020; Kong et al., 2019)

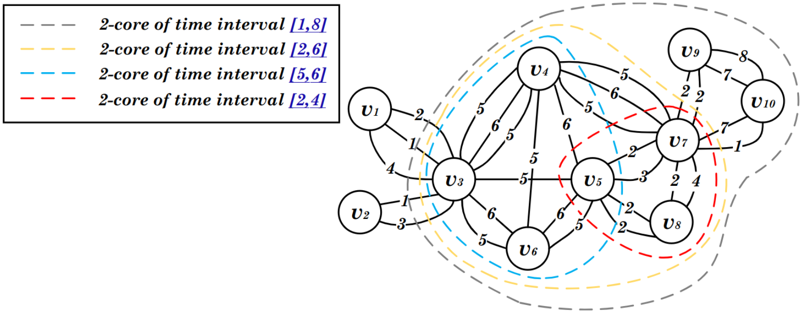
Temporal k-Core Decompositions

Variant	**Ref.**	**Running Time**	**Description**
Historical k-core	(Yu et al., 2021)	$\mathcal{O}(\log m + m_I)$	static cores spanning fixed interval I
Time-range k-core	(Yang et al., 2023)	$\mathcal{O}(\log m + \|I\| \cdot m_I)$	static cores in fixed interval
(k, h)-core	(Wu et al., 2015)	$\mathcal{O}(n + m)$	parallel temporal edges
Span-core	(Galimberti et al., 2020)	$\mathcal{O}(\|T\|^2 \cdot m)$	cores spanning intervals
(k, Δ)-core	(Oettershagen et al., 2023a)	$\mathcal{O}(m \cdot \delta_m)$	based on temporal edge degree

- more variants available (Lotito and Montresor, 2020; Hung and Tseng, 2021; Qin et al., 2022, 2020; Li et al., 2018; Oettershagen et al., 2023b) ...

- choosing the right one depends on available data and application

Historical and Time-Range k-Cores

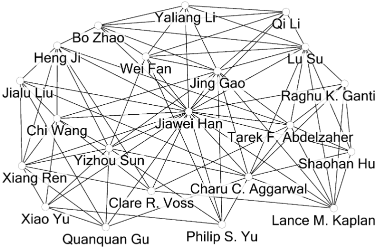


- definitions based on time window graph
  - static, aggregated graph for time window I
- historical: find a static at least k-core in time window graph for given time interval I
- time-range: find all distinct at least k-cores in all possible time windows in time interval I

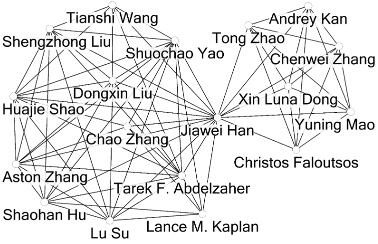
(Yu et al., 2021; Yang et al., 2023)

Historical and Time-Range k-Cores

6-cores of Prof. Jiawei Han's ego network on the DBLP snapshots



(a) *DBLP* (2011–2015)



(b) *DBLP* (2016–2020)

- straight-forward computation: restrict to interval (or subintervals)
- index-based solution to support the k-core query for every possible time window and integer k

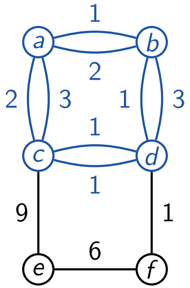
(Yu et al., 2021; Yang et al., 2023)

(k, h)-Cores

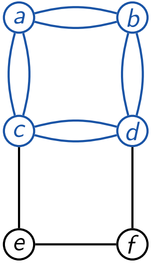
Definition

(k, h)-core is the largest subgraph H such that every v in H must have at least k neighbors in H, where each such neighbor must be connected to v with at least h temporal edges

- can be interpreted as core decomposition for multi(-layer) graphs



(a) Temporal graph G



(b) Underlying multi graph

- nodes $\{a, b, c, d\}$ induce a $(2, 2)$-core

(Wu et al., 2015)

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Applications:

- community search

(Galimberti et al., 2020; Ciaperoni et al., 2020)

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Applications:

- community search
- identify temporal patterns

(Galimberti et al., 2020; Ciaperoni et al., 2020)

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Applications:

- community search
- identify temporal patterns
- anomaly detection

(Galimberti et al., 2020; Ciaperoni et al., 2020)

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Applications:

- community search
- identify temporal patterns
- anomaly detection
- graph embedding and vertex classification

(Galimberti et al., 2020; Ciaperoni et al., 2020)

Span-Cores

Definition

- the (k, Δ)-core is a maximal set of vertices C such that C is a k-core over the complete span of time interval Δ (each edge of the core exists in each time step in Δ)
- a span-core is maximal if no other span-core dominates it in k or Δ

Applications:

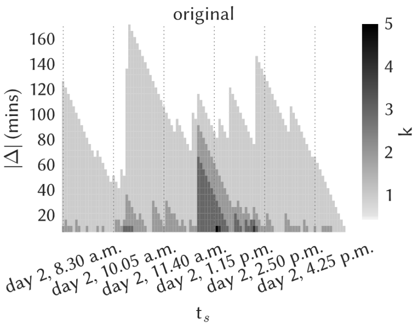
- community search
- identify temporal patterns
- anomaly detection
- graph embedding and vertex classification
- containing or maximizing spreading

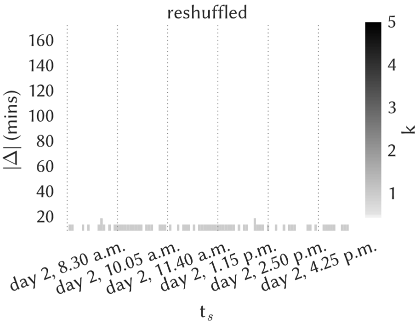
(Galimberti et al., 2020; Ciaperoni et al., 2020)

Span-Cores

Application: Temporal pattern identification

- temporal activity of a high school day



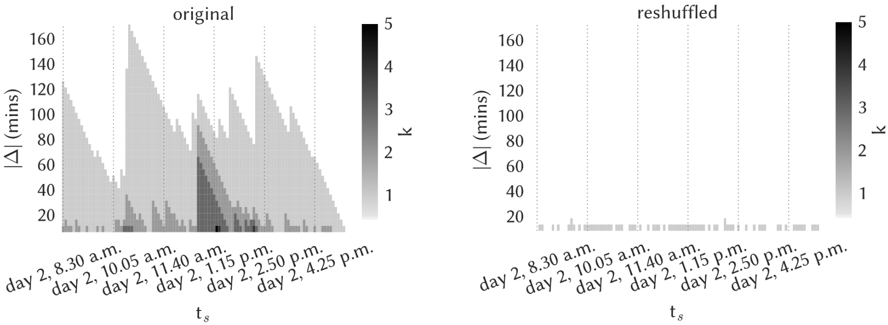


(Galimberti et al., 2020)

Span-Cores

Application: Temporal pattern identification

- temporal activity of a high school day
- span-core decomposition detects time-evolving cohesive substructures

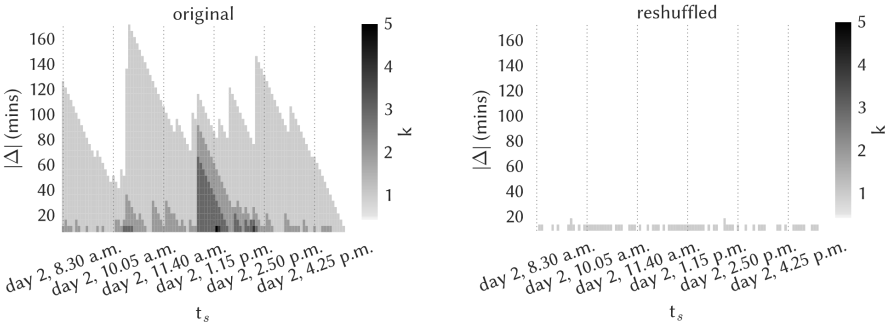


(Galimberti et al., 2020)

Span-Cores

Application: Temporal pattern identification

- temporal activity of a high school day
- span-core decomposition detects time-evolving cohesive substructures
- these completely disappear in the reshuffled data set



(Galimberti et al., 2020)

(k, Δ)-cores

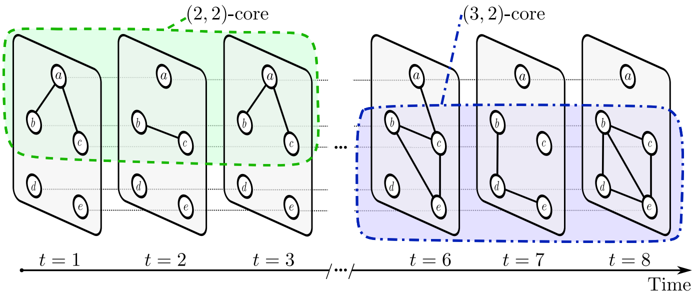
Definition

- Δ-degree of an edge is the minimum number of edges incident to one of its endpoints that have a temporal distance of at most Δ
- the (k, Δ)-core is the inclusion-maximal edge-induced subgraph $C_{(k,\Delta)}$ of G such that each temporal edge $e = (\{u, v\}, t)$ in $C_{(k,\Delta)}$ has at least a Δ-degree of $d_\Delta(e) \geq k + 1$.

(k, Δ)-cores

Definition

- Δ-degree of an edge is the minimum number of edges incident to one of its endpoints that have a temporal distance of at most Δ
- the (k, Δ)-core is the inclusion-maximal edge-induced subgraph $C_{(k,\Delta)}$ of G such that each temporal edge $e = (\{u, v\}, t)$ in $C_{(k,\Delta)}$ has at least a Δ-degree of $d_\Delta(e) \geq k + 1$.



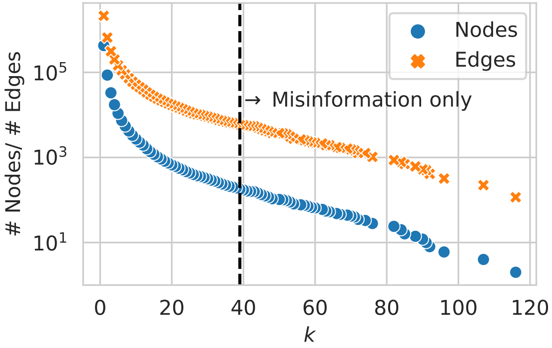
- each edge in a (k, Δ)-core is at both ends incident to at least $k + 1$ edges in the core with temporal distance at most Δ

(Oettershagen et al., 2023a)

(k, Δ)-cores

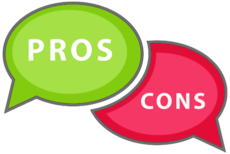
Application:

- analyzing malicious retweets in the Twitter network
- the most inner cores only contain misinformation for $\Delta = 1$ hour



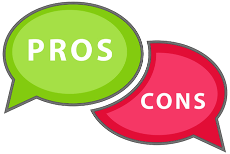
Temporal k-Core Decompositions

- **Historical and time range *k*-core**
  - based on time window graphs but efficient



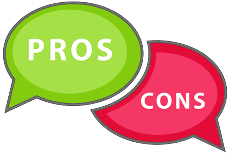
Temporal k-Core Decompositions

- **Historical and time range** k**-core**
  - based on time window graphs but efficient
- (k, h)**-core**
  - considers only multilayer graph, also efficient



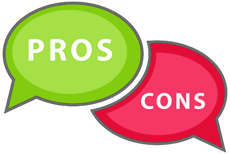
Temporal k-Core Decompositions

- **Historical and time range k-core**
  - based on time window graphs but efficient
- (k, h)-**core**
  - considers only multilayer graph, also efficient
- **Span-core**
  - needs dense sequence of static graphs, also not very scalable
  - shown to be useful in wide range of applications



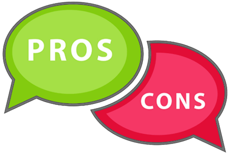
Temporal k-Core Decompositions

- **Historical and time range k-core**
  - based on time window graphs but efficient
- (k, h)**-core**
  - considers only multilayer graph, also efficient
- **Span-core**
  - needs dense sequence of static graphs, also not very scalable
  - shown to be useful in wide range of applications
- (k, Δ)**-core**
  - can handle fine-grained data and efficient



Temporal k-Core Decompositions

- **Historical and time range k-core**
  - based on time window graphs but efficient
- (k, h)-**core**
  - considers only multilayer graph, also efficient
- **Span-core**
  - needs dense sequence of static graphs, also not very scalable
  - shown to be useful in wide range of applications
- (k, Δ)-**core**
  - can handle fine-grained data and efficient



- choosing the right one depends on available data and application

Agenda

Part I : **Introduction and Motivation**

- models of temporal networks
- algorithmic approaches

Part II : **Mining Temporal Networks A:**

- connectivity, temporal properties
- centrality, cores

Part III : **Mining Temporal Networks B:**

- communities, patterns and events
- diffusion and random networks

Part IV : **Tools and Code Libraries**

Part V : **Challenges, Open Problems, and Trends**

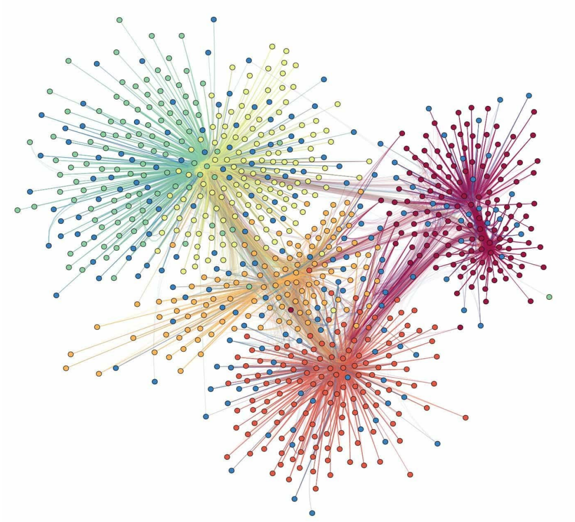
Part IV

Mining Temporal Networks B

Community Detection

Temporal Communities

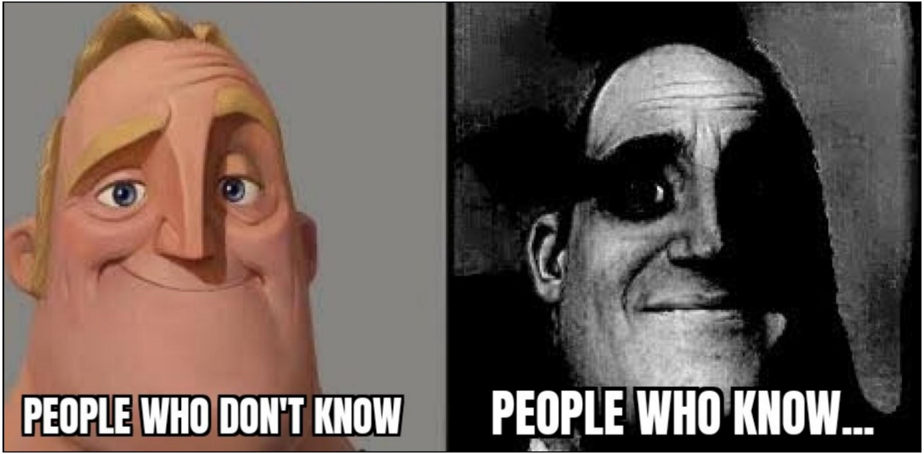
Identifying communities is a fundamental task in *computer* and *network* science.



Temporal Communities

Question: How do we define a temporal community?

Question: How do we define a temporal community?



Communities in Static Networks

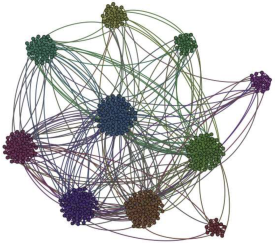
"community" = "umbrella term"

- extensive surveys (Fortunato and Hric, 2016; Su et al., 2024)
- possible definitions
  - a community is a set of nodes, closer to each other than to the rest of the network
  - a community is a "dense" subgraph

Communities in Static Networks

"community" = "umbrella term"

- extensive surveys (Fortunato and Hric, 2016; Su et al., 2024)
- possible definitions
  - a community is a set of nodes, closer to each other than to the rest of the network
  - a community is a "dense" subgraph
- *typical* problem settings
  - a single community vs. network partitioning
  - overlapping vs. non-overlapping communities
  - local to some nodes vs. global



Community Detection in Static Network

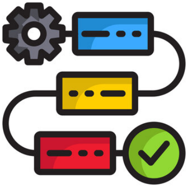
Usual workflow (data analysis)

1. pick a problem setting (e.g., partition in k communities vs identify a single local community)

2. pick a proper metric to quantify the "density" of the community S
   - average degree : $\frac{|E(S)|}{2|S|}$
   - density : $\frac{2|E(S)|}{|S|(|S|-1)}$
   - conductance : $\frac{cut(S,\bar{S})}{\min\{vol(S),vol(\bar{S})\}}$
   - modularity
   - ...

3. identify/design proper algorithms to solve the problem

4. analyze data and, if needed repeat from steps 1. or 2.



Analyses and Applications

- social networks
  - link prediction, targeted advertisement, content moderation, etc..

Analyses and Applications

- social networks
  - link prediction, targeted advertisement, content moderation, etc..
- financial networks
  - fraud detection, money-laundry activities, etc..

Analyses and Applications

- social networks
  - link prediction, targeted advertisement, content moderation, etc..
- financial networks
  - fraud detection, money-laundry activities, etc..
- collaboration networks
  - identifying trending topics, important group of nodes, etc..
- ...

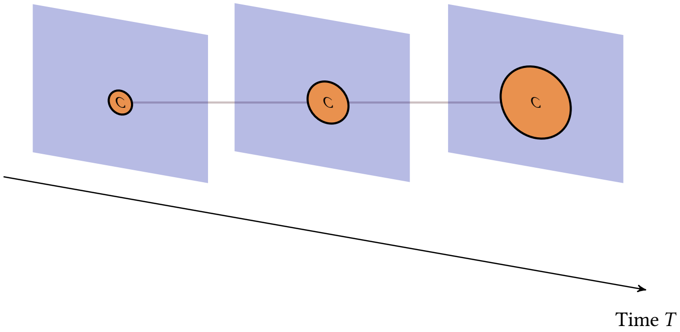
So what is new about temporal communities?

Temporal Evolution of Communities

Temporal networks allow us to study communities according to their **temporal evolution**!

Some representative behaviors,

Growing community C



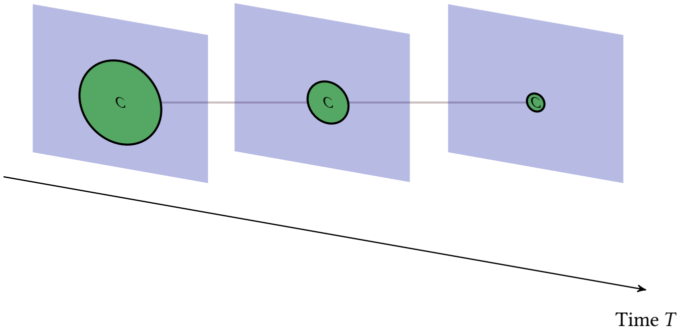
Temporal Evolution of Communities

Temporal networks allow us to study communities according to their **temporal evolution**!

Some representative behaviors,

Growing community C

Shrinking community C



Temporal Evolution of Communities

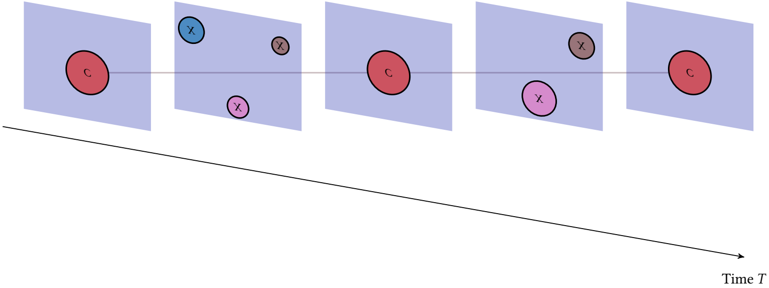
Temporal networks allow us to study communities according to their **temporal evolution**!

Some representative behaviors,

Growing community C

Shrinking community C

Periodic community C



Temporal Evolution of Communities

temporal networks allow us to study communities according to their **temporal evolution**!

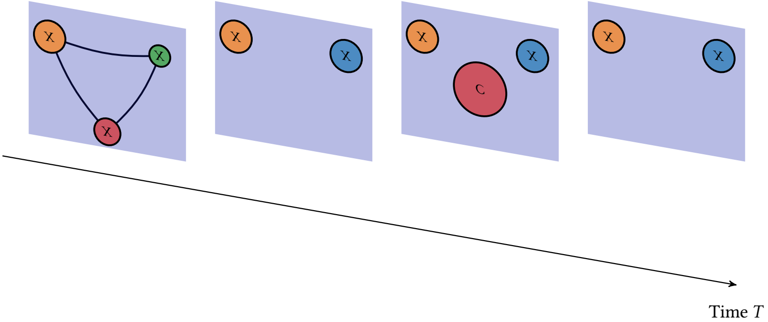
Some representative behaviors,

Growing community C

Shrinking community C

Periodic community C

Bursty community C

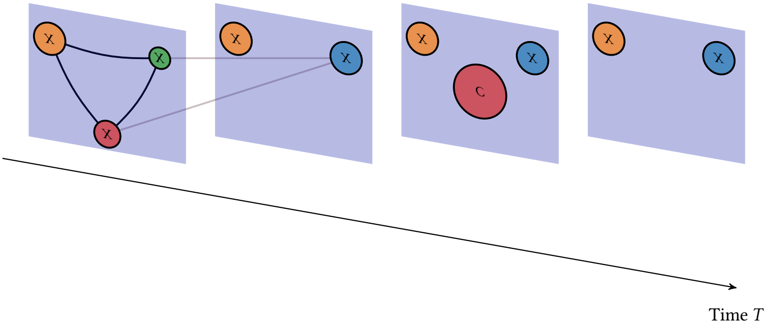


Temporal Evolution of Communities

Temporal networks allow us to study communities according to their **temporal evolution**!

Some representative behaviors,

- Growing community C
- Shrinking community C
- Periodic community C
- Bursty community C
- Merging communities
- ...



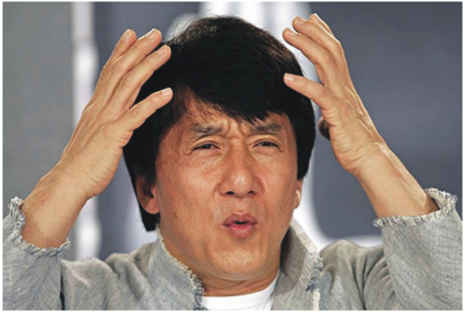
Are there proposed taxonomies?

Community Detection in Temporal Networks

Question How many taxonomies exist?

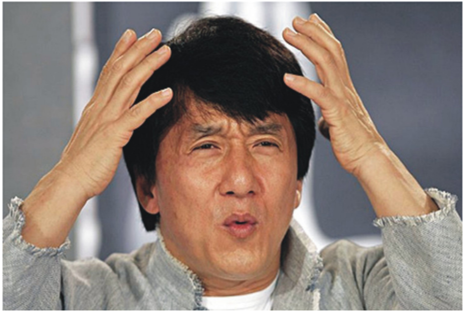
Community Detection in Temporal Networks

- proposed taxonomies
  - (Aynaud et al., 2013)
  - (Aggarwal and Subbian, 2014)
  - (Enugala et al., 2015)
  - (Renaud and Naoki, 2016)
  - (Hartmann et al., 2016)
  - (Rossetti and Cazabet, 2018)
  - (Dakiche et al., 2019)
  - (Christopoulos and Tsichlas, 2022)
  - ...



Community Detection in Temporal Networks

- proposed taxonomies
  - (Aynaud et al., 2013)
  - (Aggarwal and Subbian, 2014)
  - (Enugala et al., 2015)
  - (Renaud and Naoki, 2016)
  - (Hartmann et al., 2016)
  - (Rossetti and Cazabet, 2018)
  - (Dakiche et al., 2019)
  - (Christopoulos and Tsichlas, 2022)
  - ...

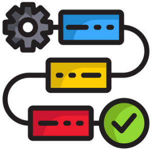


No need to panic: recall our steps for community identification!

Temporal Community Detection

Proposed workflow (exploratory analysis), everything starts from data!

1. identify if *temporal* data is fine-grained or course-grained (!)
2. pick a problem setting (✓)
3. pick a proper metric to quantify the "**temporal** *density*" of the community S, encoding the desired temporal properties (!)
4. identify/design proper algorithms to solve the problem
5. analyze data and, if needed repeat from steps 1. or 2.



Lets make it more concrete!

BFF: Finding Lasting Communities (Semertzidis et al., 2016)

Data. Consider a temporal network $G = \{G_1, \ldots, G_T\}$

Setting. We want to find a single global *dense* community across all snapshots!

Implicit assumption. time is sufficiently course-grained so each snapshot has enough information

Metrics and temporal properties. Fix a time t, given $S \subseteq V$ we define,

- $d_{avg}(S, G_t) = \frac{1}{|S|} \sum_{u \in S} d(u, G_t[S]) = \frac{2|E(S, G_t)|}{|S|}$, and
- $d_{min}(S, G_t) = \min_{u \in S} d(u, G_t[S])$

BFF: Finding Lasting Communities (Semertzidis et al., 2016)

Data. Consider a temporal network $G = \{G_1, \ldots, G_T\}$

Setting. We want to find a single global *dense* community across all snapshots!

Implicit assumption. time is sufficiently course-grained so each snapshot has enough information

Metrics and temporal properties. Fix a time t, given $S \subseteq V$ we define,

- $d_{avg}(S, G_t) = \frac{1}{|S|} \sum_{u \in S} d(u, G_t[S]) = \frac{2|E(S,G_t)|}{|S|}$, and
- $d_{min}(S, G_t) = \min_{u \in S} d(u, G_t[S])$

Combining such values across snapshots, let $- \in \{avg, min\}$

- $g_{min}(d_-(S, G)) = \min_{t=1,\ldots,T} d_-(S, G_t)$
- $g_{avg}(d_-(S, G)) = \frac{1}{T} \sum_t d_-(S, G_t)$

BFF: Finding Lasting Communities (Semertzidis et al., 2016)

Data. Consider a temporal network $G = \{G_1, \ldots, G_T\}$

Setting. We want to find a single global *dense* community across all snapshots!

Implicit assumption. time is sufficiently course-grained so each snapshot has enough information

Metrics and temporal properties. Fix a time t, given $S \subseteq V$ we define,

- $d_{avg}(S, G_t) = \frac{1}{|S|} \sum_{u \in S} d(u, G_t[S]) = \frac{2|E(S, G_t)|}{|S|}$, and
- $d_{min}(S, G_t) = \min_{u \in S} d(u, G_t[S])$

Combining such values across snapshots, let $- \in \{avg, min\}$

- $g_{min}(d_-(S, G)) = \min_{t=1,\ldots,T} d_-(S, G_t)$
- $g_{avg}(d_-(S, G)) = \frac{1}{T} \sum_t d_-(S, G_t)$

So we finally have a score for a community $S \subseteq V$, that is given $+, - \in \{avg, min\}$ we let

$$f_{+,-}(S, G) = g_+(d_-(S, G))$$

BFF: A Closer Look on the Metrics

Problem

Given $G = (G_1, \ldots, G_T)$, let $+, - \in \{avg, min\}$ and a target density $f_{+,-}$ find a subset of vertices $S^* \subseteq V$ **maximizing** the objective $f_{+,-}(S, G)$ over all communities.

Inspecting the objective values

BFF: A Closer Look on the Metrics

Problem

Given $G = (G_1, \ldots, G_T)$, let $+, - \in \{avg, min\}$ and a target density $f_{+,-}$ find a subset of vertices $S^* \subseteq V$ **maximizing** the objective $f_{+,-}(S, G)$ over all communities.

Inspecting the objective values

- $f_{min,min}(S, G) = g_{min}(d_{min}(S, G)) = \min_{t=1,\ldots,T} \min_{u \in S} d(u, G_t[S]))$,
  minimum degree on *every* snapshot of *all* vertices $v \in S^*$ is high
- $f_{min,avg}(S, G) = g_{min}(d_{avg}(S, G)) = \min_{t=1,\ldots,T} \frac{2|E(S,G_t)|}{|S|}$,
  average degree of each node in S is large on each snapshot
- $f_{avg,min}(S, G) = g_{avg}(d_{min}(S, G)) = \frac{1}{T} \sum_t \min_{u \in S} d(u, G_t[S]))$,
  on average the minimum degree of each node in S is high (more flexible than $f_{min,min}$)
- $f_{avg,avg}(S, G) = g_{avg}(d_{avg}(S, G)) = \frac{1}{T} \sum_t \frac{2|E(S,G_t)|}{|S|}$,
  on average the average degree of each node in S is high

BFF: Algorithm

Computing the solution, FindBFF (Inspired by Charikar (2000))

1. iteratively and greedily peel V removing at each step $v = \arg\min_{v \in V} score(v, G[V])$
2. compute the density target density on the remaining network where $V = V \setminus \{v\}$
3. return the vertex-set $S \subseteq V$ maximizing the target density over all $\mathcal{O}(n)$ iterations

BFF: Algorithm

Computing the solution, FindBFF (Inspired by Charikar (2000))

1. iteratively and greedily peel V removing at each step $v = \arg\min_{v \in V} score(v, G[V])$
2. compute the density target density on the remaining network where $V = V \setminus \{v\}$
3. return the vertex-set $S \subseteq V$ maximizing the target density over all $\mathcal{O}(n)$ iterations

How is $score(v, G[V])$ computed? Depends on the objective f

- for $f_{min,min}(S, G)$ we set $score(v, G[V_i]) = \min_{t=1,\dots,T} d(v, G_t[V])$
- for $f_{avg,avg}(S, G)$ we set $score(v, G[V_i]) = \frac{1}{T} \sum_t d(v, G_t[V])$

BFF: Algorithm

Computing the solution, FindBFF (Inspired by Charikar (2000))

1. iteratively and greedily peel V removing at each step $v = \arg\min_{v \in V} score(v, G[V])$
2. compute the density target density on the remaining network where $V = V \setminus \{v\}$
3. return the vertex-set $S \subseteq V$ maximizing the target density over all $\mathcal{O}(n)$ iterations

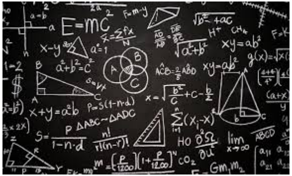
How is $score(v, G[V])$ computed? Depends on the objective f

- for $f_{min,min}(S, G)$ we set $score(v, G[V_i]) = \min_{t=1,\dots,T} d(v, G_t[V])$
- for $f_{avg,avg}(S, G)$ we set $score(v, G[V_i]) = \frac{1}{T} \sum_t d(v, G_t[V])$

Resulting algorithms run in $\mathcal{O}(nT + \sum_t |E_t|)$

- FindBFF-MM ($f_{min,min}$), finds the optimal solution
- FindBFF-AA ($f_{avg,avg}$), finds a $\frac{1}{2}$-approximation
- BFF-AM ($f_{avg,min}$) is **NP**-hard, FindBFF achieves an approximation at most $\mathcal{O}(n^{-1})$
- BFF-MA ($f_{min,avg}$), "complexity = ?", FindBFF achieves an approximation at most $\mathcal{O}(n^{-1/2})$

On the Complexity of BFF-MA



On this line of research Charikar et al. (2018) proved

- $f_{min,avg}(S, G)$ cannot be approximated within $2^{\log^{1-\varepsilon} n}$ unless $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{poly \log(n)})$
- they give $\mathcal{O}((n \log T)^{-1/2})$ and $\mathcal{O}(n^{-2/3})$ approximation algorithms
- suppose T is small the authors give and exact algorithm running in $\mathcal{O}(n^T poly(n, T))$, and a FPTAS that given an $\varepsilon > 0$ outputs a $(1 + \varepsilon)$-approximation in $\mathcal{O}(f(T) poly(n, \varepsilon^{-1}))$.

FindBFF — A Use Case

Dataset consists of publications from DBLP in years from 2006 to 2015 (each year forms a snapshot G_i), the set of nodes V ($|V| = 2625$) represents authors.

FindBFF — A Use Case

Dataset consists of publications from DBLP in years from 2006 to 2015 (each year forms a snapshot G_i), the set of nodes V ($|V| = 2625$) represents authors.

BFF-MM	BFF-MA	BFF-AM	BFF-AA
Wei Fan, Philip S. Yu, Jiawei Han, Charu C. Aggarwal Lu Qin, Jeffrey Xu Yu, Xuemin Lin Craig Macdonald, Iadh Ounis	*Wei Fan*, Jing Gao, *Philip S. Yu, Jiawei Han, Charu C. Aggarwal* Jeffrey Xu Yu, Xuemin Lin, Ying Zhang	*Wei Fan*, Jing Gao, *Philip S. Yu, Jiawei Han*	*Wei Fan, Jing Gao, Philip S. Yu, Jiawei Han, Charu C. Aggarwal,* Mohammad M. Masud, Latifur Khan, Bhavani M. Thuraisingham

Some observations

- Not all authors appearing in a dense subset coauthored many papers together, e.g., "Wei Fan", "Philip S. Yu", and "Jiawei Han" coauthored only two papers together but many pairwise.
- Some solutions are *not connected*, e.g., BFF-MM.

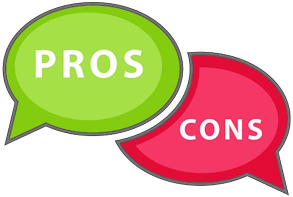
Pros and Cons

Strengths

- computes dense communities based on degree scores
- it is efficient for certain formulations
- works well for sufficiently temporal course-grained data

What is missing

- not optimizing for specific time-frames of the time-span
- cannot adapt well for fine-grained data
- theoretical gaps may prevent (close to) optimal solutions



Lets see *another formulation*!

Communities as Episodes (Rozenshtein et al., 2019)

Data. Let $G = (V, E)$ be an undirected temporal graph with $E = \{(u_i, v_i, t_i) : i = 1, \ldots, m\}$.

Setting. Find a multiple *dense* communities covering the timespan $[t_1, t_m]$

Communities as Episodes (Rozenshtein et al., 2019)

Data. Let $G = (V, E)$ be an undirected temporal graph with $E = \{(u_i, v_i, t_i) : i = 1, \ldots, m\}$.

Setting. Find a multiple *dense* communities covering the timespan $[t_1, t_m]$

Metrics and temporal properties.
Given an interval $I = [t_s, t_e] \subseteq [t_1, t_m]$ let $G[I] = (V[I], E[I])$ be the induced subgraph in I

An *episode* is a pair $(I = [t_s, t_e], H \subseteq G[I])$ where H is a subgraph of $G[I]$

Given a static subgraph $H = (V_H, E_H)$ its density is $d(H) = \frac{|E_H|}{|V_H|}$

Communities as Episodes (Rozenshtein et al., 2019)

Data. Let $G = (V, E)$ be an undirected temporal graph with $E = \{(u_i, v_i, t_i) : i = 1, \ldots, m\}$.

Setting. Find a multiple *dense* communities covering the timespan $[t_1, t_m]$

Metrics and temporal properties.
Given an interval $I = [t_s, t_e] \subseteq [t_1, t_m]$ let $G[I] = (V[I], E[I])$ be the induced subgraph in I

An *episode* is a pair $(I = [t_s, t_e], H \subseteq G[I])$ where H is a subgraph of $G[I]$

Given a static subgraph $H = (V_H, E_H)$ its density is $d(H) = \frac{|E_H|}{|V_H|}$

Problem

Given a temporal network G and an integer k find a set of k episodes $(I_i, H_i), i = 1, \ldots, k$ such that $I_i \cap I_j = \emptyset, i \neq j$ are disjoint while maximizing

$$\sum_{i=1}^{k} d(H_i)$$

Solving the Problem trough Segmentation

Computing the solution

- **Note!** in an optimal solution $\cup_i I_i$ covers $[t_1, t_m]$.
- given interval I use result in (Goldberg, 1984) in $O(nm_{st} \log n)$ to obtain $H^* = \arg\max_{H \subseteq G[I]} d(H)$, and m_{st} is the max edges in an interval
- it suffices to find an optimal segmentation $I_1, \ldots, I_k$, (with dyn. prog. in $\mathcal{O}(kT^2)$)

Such algorithm has runtime $\mathcal{O}(kT^2 nm_{st} \log n)$

Solving the Problem trough Segmentation

Computing the solution

- **Note!** in an optimal solution $\cup_i I_i$ covers $[t_1, t_m]$.
- given interval I use result in (Goldberg, 1984) in $O(nm_{st} \log n)$ to obtain $H^* = \arg\max_{H \subseteq G[I]} d(H)$, and m_{st} is the max edges in an interval
- it suffices to find an optimal segmentation $I_1, \ldots, I_k$, (with dyn. prog. in $\mathcal{O}(kT^2)$)

Such algorithm has runtime $\mathcal{O}(kT^2 nm_{st} \log n)$

As this can be prohibitive an approximate approach is proposed

- *approximate dyn. prog.* approach controlled ε_1 pruning candidates in segmentation
- avoiding recomputing densest subgraph, use *evolving solution*, controlled by ε_2

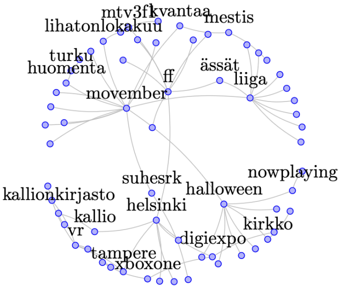
Leading to $2(1+\varepsilon_1)(1+\varepsilon_2)$-approximation algorithm running in $\mathcal{O}(\frac{k^2}{\varepsilon_1\varepsilon_2} T m_{T,max} \log^2(n))$ where $m_{T,max}$ is the maximum number of edges with the same time-stamp.

Communities as Episodes — Analyzing Twitter Data

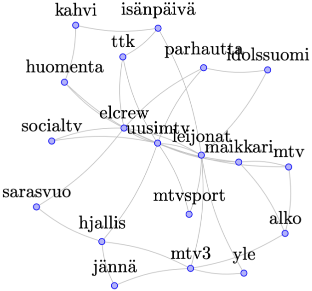
Tweets in Helsinki region, V are hashtags and (u, v, t) is added if a message contains both hashtags u, v, data gathered from Nov 2013 containing 4758 tweets and 917 nodes. (Parameters, $k = 4$ and $\varepsilon_1 = \varepsilon_2 = 0.1$)

Communities as Episodes — Analyzing Twitter Data

Tweets in Helsinki region, V are hashtags and (u, v, t) is added if a message contains both hashtags u, v, data gathered from Nov 2013 containing 4758 tweets and 917 nodes. (Parameters, $k = 4$ and $\varepsilon_1 = \varepsilon_2 = 0.1$)



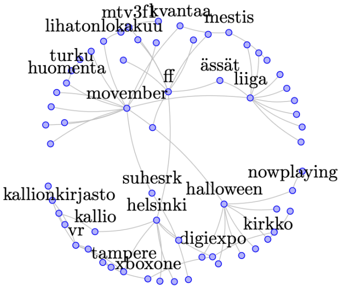
01.11 03:31 – 02.11 11:00



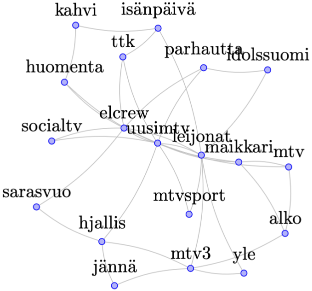
02.11 11:01 – 11.11 12:53

Communities as Episodes — Analyzing Twitter Data

Tweets in Helsinki region, V are hashtags and (u, v, t) is added if a message contains both hashtags u, v, data gathered from Nov 2013 containing 4758 tweets and 917 nodes. (Parameters, $k = 4$ and $\varepsilon_1 = \varepsilon_2 = 0.1$)



01.11 03:31 – 02.11 11:00



02.11 11:01 – 11.11 12:53

- both episodes are related to events occurring in that period in Helsinki and globally, e.g., on the left the Digiexpo and Halloween events, while on the right the MTV Europe music awards that was on November 10.

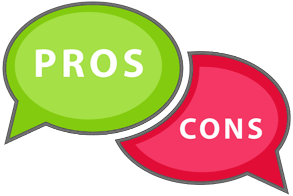
Pros and Cons

Strengths

- it adapts in a flexible way to the timespan of the network
- in general it is efficient to compute
- identifies dense communities (w.r.t., average degree)

What is missing

- when aggregation is performed may lose some information
- not accounting for temporal behaviors, e.g., periodicity or burstiness
- may not be informative with small timespan (i.e, G has not few dense structures)



Lets see another formulation!


What about
Social Networks

Discovering Buzzing Stories (Bonchi et al., 2019)

Data. Given a temporal network $G = (V, \{E_t, f_t\}_{t=1,\ldots,T})$.

Discovering Buzzing Stories (Bonchi et al., 2019)

Data. Given a temporal network $G = (V, \{E_t, f_t\}_{t=1,\dots,T})$.

- V is a set of objects, e.g., *messages or words* from chat apps or posts on socials.

- E_t edges at time t and $f_t : E_t \mapsto \mathbb{R}^+$ is a weighting function capturing the strength between two objects at t, e.g., $f_t(\text{cream}, \text{carbonara})$

Discovering Buzzing Stories (Bonchi et al., 2019)

Data. Given a temporal network $G = (V, \{E_t, f_t\}_{t=1,\dots,T})$.

Setting. Find multiple *dense* and unexpected *stories*, i.e., communities.

Metrics and temporal properties.

Transform $G \to G_A = (V, \{E_t, \phi_t\}_{t=1,\dots,T})$ by mapping $f_t \to \phi_t$ where $\phi_t(\cdot)$ captures how anomalous or unexpected is e based on past data.

Given a *discrete*-time interval $I = [t_s, t_e]$ and $S \subseteq G_A$ then "$\delta(S, I)$ = density of S in I"

Given a set of subgraphs $\mathcal{S}$ then $\Delta(\mathcal{S}, I) = \sum_{S \in \mathcal{S}} \delta(S, I)$.

Discovering Buzzing Stories (Bonchi et al., 2019)

Data. Given a temporal network $G = (V, \{E_t, f_t\}_{t=1,\dots,T})$.

Setting. Find multiple *dense* and unexpected *stories*, i.e., communities.

Metrics and temporal properties.

Transform $G \rightarrow G_A = (V, \{E_t, \phi_t\}_{t=1,\dots,T})$ by mapping $f_t \rightarrow \phi_t$ where $\phi_t(\cdot)$ captures how anomalous or unexpected is e based on past data.

Given a *discrete*-time interval $I = [t_s, t_e]$ and $S \subseteq G_A$ then "$\delta(S, I) =$ density of S in I"

Given a set of subgraphs $\mathcal{S}$ then $\Delta(\mathcal{S}, I) = \sum_{S \in \mathcal{S}} \delta(S, I)$.

Problem

Given the temporal (anomalous) graph G_A, an interval $I \subseteq [t_1, t_m]$ and two integers $K, N \geq 1$ find $\mathcal{S}^* = \{S_1, \dots, S_K\}$ of disjoint subgraphs of G_A such that

- $|S_i| \leq N$, that is the number of nodes in each subgraph is bounded by N
- $\Delta(\mathcal{S}^*, I)$ is maximized.

Discovering Buzzing Stories — Algorithm

The decision problem is **NP**-hard

Computing a solution

- develop an *exact* algorithm $\mathcal{A}$ running in $\mathcal{O}(|I| m \log n)$ for $K = 1$ and $N = \infty$ based on a "peeling" the minimum degree vertices.

  Note! This needs to be accounting for time-points (updating the vertex degrees).

Discovering Buzzing Stories — Algorithm

The decision problem is **NP**-hard

Computing a solution

- develop an *exact* algorithm $\mathcal{A}$ running in $\mathcal{O}(|I|m \log n)$ for $K = 1$ and $N = \infty$ based on a "peeling" the minimum degree vertices.

  Note! This needs to be accounting for time-points (updating the vertex degrees).

- algorithm for general case using $\mathcal{A}$ as subroutine by
  - imposing further *constraints on the size* (i.e., bounding N) of the reported solutions
  - iteratively *remove identified communities* to guarantee disjoint output.

The resulting algorithms runs in $\mathcal{O}(K|I|m \log n)$ and comes without guarantees.

Discovering Buzzing Stories – A Use Case

Dataset. Yahoo searches during 2013-2014

Assumption. If there is an anomaly people will search it on the web!

Processing.

- dataset spans 558 days, is build on user queries (appearing at least 50 times per day)
- map queries e.g., "How to put pineapple on pizza" $\rightarrow$ "(pineapple,pizza)" is generated
- f_t accounts for frequency

Discovering Buzzing Stories – A Use Case

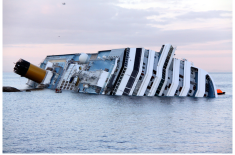
Dataset. Yahoo searches during 2013-2014

Assumption. If there is an anomaly people will search it on the web!

Processing.

- dataset spans 558 days, is build on user queries (appearing at least 50 times per day)
- map queries e.g., "How to put pineapple on pizza" $\rightarrow$ "(pineapple,pizza)" is generated
- f_t accounts for frequency

Date	$\|I\|$	N	**Story**	**Related Event**
13/01/2014	1	10	"cristiano dor wins ronaldo fifa ballon"	Cristiano Ronaldo won the Ballon d'Or in 2014
09/02/2014	3	10	"day figure russia julia skating medal ceremony"	Yulia Lipnitskaya, a Russian prodigy won gold medal in Sochi
27/02/2014	2	30	"captains costa wreck concordia"	Legal process for the Costa Concordia disaster



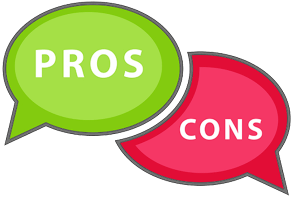
Pros and Cons

Strengths

- tailored to a specific application
- interesting analyses
- efficient algorithm in practice

What is missing

- no guarantees
- a lot of preprocessing is needed and identifying $\phi(\cdot)$ may be non-trivial
- not much used in practice




TIME FOR...
QUESTIONS

Significant Engagement Based Community (Zhang et al., 2022)

Dataset. Undirected temporal network $G = (V, E), E = \{(u_i, v_i, t_i) : i = 1, \ldots, m\}$

Setting. Find the community where a given **user** has highest engagement, *local formulation*!

Significant Engagement Based Community (Zhang et al., 2022)

Dataset. Undirected temporal network $G = (V, E), E = \{(u_i, v_i, t_i) : i = 1, \dots, m\}$

Setting. Find the community where a given **user** has highest engagement, *local formulation*!

Metrics and temporal properties.
Let $H \subseteq G$, the *degree* of $v \in V[H]$ is $d_{u,H} = \sum_{e:E[H]} \mathbf{1}[v \in e]$.

Given $H \subseteq G$ the *engagement* of $v \in V[H]$ is $\gamma(u, H) = \frac{d_{u,H}}{\sum_{v \in V[H]} d_{v,H}}$.

Significant Engagement Based Community (Zhang et al., 2022)

Dataset. Undirected temporal network $G = (V, E), E = \{(u_i, v_i, t_i) : i = 1, \dots, m\}$

Setting. Find the community where a given **user** has highest engagement, *local formulation*!

Metrics and temporal properties.
Let $H \subseteq G$, the *degree* of $v \in V[H]$ is $d_{u,H} = \sum_{e:E[H]} \mathbf{1}[v \in e]$.

Given $H \subseteq G$ the *engagement* of $v \in V[H]$ is $\gamma(u, H) = \frac{d_{u,H}}{\sum_{v \in V[H]} d_{v,H}}$.

Problem

Given a temporal graph G, a parameter $k \geq 1$, and a vertex $u \in V$ find H such that

- $u \in V[H]$
- the static network of H is a k-core (Guarantees cohesiveness)
- $\gamma(u, H) \geq \gamma(u, H')$ for any other $H' \subseteq G$ (Guarantees max-engagement)

A Case Study on DBLP (Zhang et al., 2022)

Computing a solution

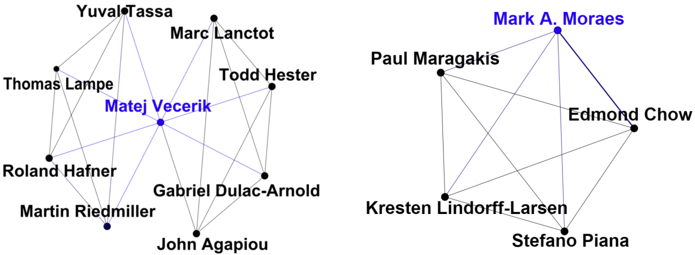
Greedy *peeling* algorithms + local search running in $\mathcal{O}(m^2(n+m))$

A Case Study on DBLP (Zhang et al., 2022)

Computing a solution

Greedy *peeling* algorithms + local search running in $\mathcal{O}(m^2(n+m))$

Insights on DBLP data



As desired seed nodes are well centered in the identified communities

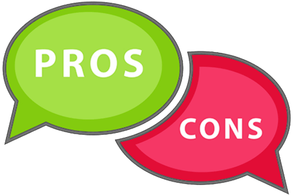
Pros and Cons

Strengths

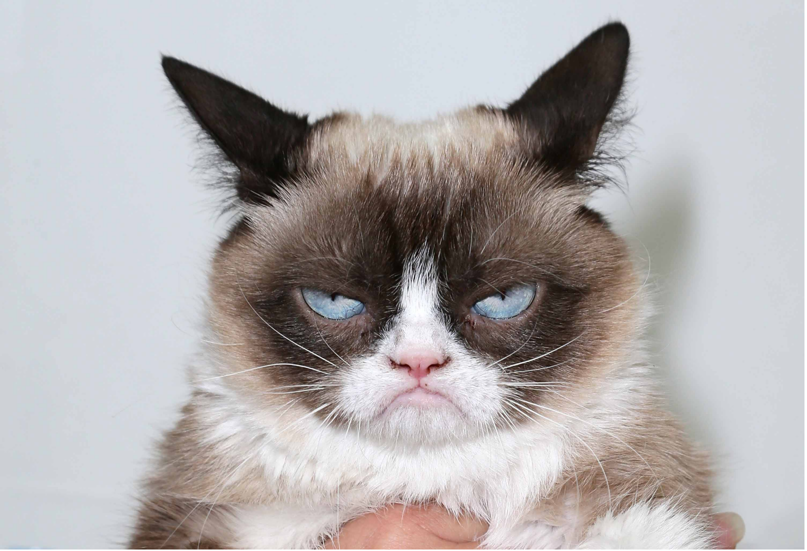
- local formulation
- proposed algorithm is polynomial
- output has desired properties

What is missing

- no guarantees
- may need additional assumptions to better model engagement
- engagement is not time-dependent



You showed us only degree-based metrics!



Local Motif Clustering (Fu et al., 2020)

Data. Undirected temporal network $G = \{G_1, \ldots, G_T\}$.

Setting. Find a good local "*tight cluster* $\to$ *motifs*" to a *seed node* at each time point $1, \ldots, T$.

Local Motif Clustering (Fu et al., 2020)

Data. Undirected temporal network $G = \{G_1, \ldots, G_T\}$.

Setting. Find a good local "*tight cluster* $\rightarrow$ *motifs*" to a *seed node* at each time point $1, \ldots, T$.

Metrics and temporal properties.

H: a small subgraph pattern (e.g., edge, triangle, star etc).

$C \subseteq V$: is cluster, and *motif-conductance* is $\Phi(C, H) = \frac{\partial(C,H)}{\min\{vol(C,H), vol(\bar{C},H)\}}$ where,

- $\partial(C, H)$: $\#$ of subgraphs *cut* by the cluster (i.e., "cut $=$ at least one node not in C")
- $vol(A, H)$: $\#$ of occurrences of H in A

Local Motif Clustering (Fu et al., 2020)

Data. Undirected temporal network $G = \{G_1, \dots, G_T\}$.

Setting. Find a good local "*tight cluster* $\rightarrow$ *motifs*" to a *seed node* at each time point $1, \dots, T$.

Metrics and temporal properties.

H: a small subgraph pattern (e.g., edge, triangle, star etc).

$C \subseteq V$: is cluster, and *motif-conductance* is $\Phi(C, H) = \frac{\partial(C,H)}{\min\{vol(C,H), vol(\bar{C},H)\}}$ where,

- $\partial(C, H)$: $\#$ of subgraphs *cut* by the cluster (i.e., "cut = at least one node not in C")
- $vol(A, H)$: $\#$ of occurrences of H in A

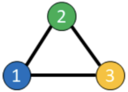
Problem

Given $G = \{G_1, \dots, G_T\}$, a static motif H, a seed node $v \in V$, an upper-bound on the motif conductance ϕ, compute,

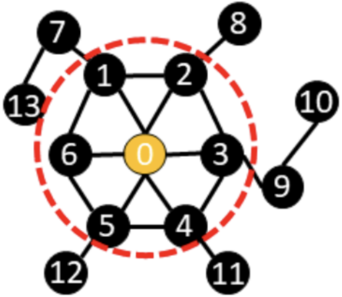
$$C^t \text{ containing } v \in V \text{ such that } \Phi(C^t, H) \leq \phi, \text{ for each } t = 1, \dots, T.$$

Local Motif Clustering – Example

Example

Let us fix H to be a triangle 

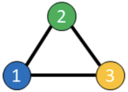
Notation. Blue edges denote insertions and green edges denote removals.



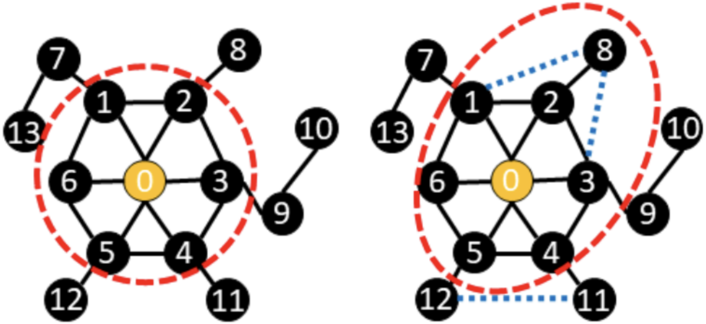
(a) $t = 1$

Local Motif Clustering – Example

Example

Let us fix H to be a triangle 

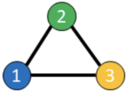
Notation. Blue edges denote insertions and green edges denote removals.



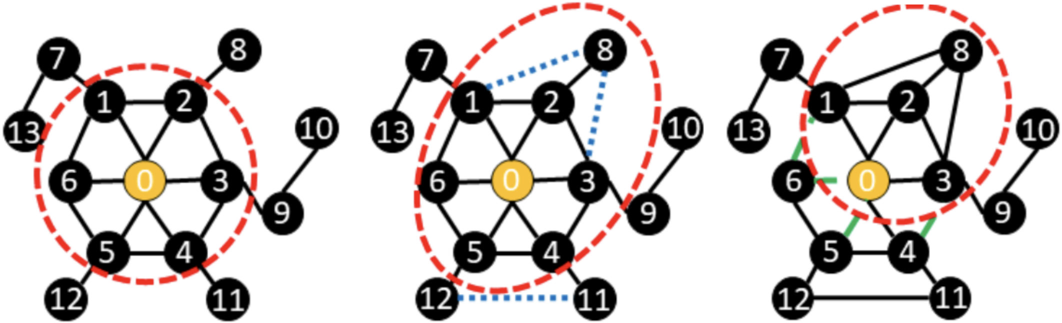
(a) $t = 1$ (b) $t = 2$

Local Motif Clustering – Example

Example

Let us fix H to be a triangle 

Notation. Blue edges denote insertions and green edges denote removals.



(a) $t = 1$ (b) $t = 2$ (c) $t = 3$

Proposed Solution by Fu et al. (2020)

The problem is already for static graphs **NP**-hard 🙁

Proposed Solution by Fu et al. (2020)

The problem is already for static graphs **NP**-hard 🙁

We will be looking for an approximate solution

Proposed Solution by Fu et al. (2020)

The problem is already for static graphs **NP**-hard 🙁

Computing a solution

- fix $t = 1, \ldots, T$, let k be $\#$ nodes in H
- define a multilinear page-rank vector $\mathbf{x}^t$, that accounts for the high-order motifs,

$$\mathbf{x}^t = \alpha P^t(\underbrace{\mathbf{x}^t \otimes \cdots \otimes \mathbf{x}^t}_{k-1 \text{ times}}) + (1-\alpha)\mathbf{u}$$

where P^t encodes transitions over motifs, $\mathbf{u}$ is the vector encoding user preferences.

Proposed Solution by Fu et al. (2020)

The problem is already for static graphs **NP**-hard 🙁

Computing a solution

- fix $t = 1, \ldots, T$, let k be $\#$ nodes in H
- define a multilinear page-rank vector $\mathbf{x}^t$, that accounts for the high-order motifs,

$$\mathbf{x}^t = \alpha P^t(\mathbf{x}^t \otimes \cdots \otimes \mathbf{x}^t) + (1 - \alpha)\mathbf{u}$$

  where P^t encodes transitions over motifs, $\mathbf{u}$ is the vector encoding user preferences.
- a good-approximate obtained through sweep cut on vector $\mathbf{x}^t$,
  - sort (from largest to smallest) the entries in $\mathbf{x}^t$
  - pick the prefix $1, \ldots, j$ with $j = 1, \ldots, n$ optimizing the induced cut

Proposed Solution by Fu et al. (2020)

The problem is already for static graphs **NP**-hard

Computing a solution

- fix $t = 1, \ldots, T$, let k be $\#$ nodes in H
- define a multilinear page-rank vector $\mathbf{x}^t$, that accounts for the high-order motifs,

$$\mathbf{x}^t = \alpha P^t(\mathbf{x}^t \otimes \cdots \otimes \mathbf{x}^t) + (1-\alpha)\mathbf{u}$$

  where P^t encodes transitions over motifs, $\mathbf{u}$ is the vector encoding user preferences.
- a good-approximate obtained through sweep cut on vector $\mathbf{x}^t$,
  - sort (from largest to smallest) the entries in $\mathbf{x}^t$
  - pick the prefix $1, \ldots, j$ with $j = 1, \ldots, n$ optimizing the induced cut

Idea! For varying t, *graph is evolving* avoid repeating such steps by **1.** avoid considering distant edges that cannot impact cluster **2.** avoid scratch re computation of $\mathbf{x}^t$

Runtime $\mathcal{O}(\sum_t [f(m_i, n_i^{O(k)}, k^k)]$, P^1 is assumed in input!

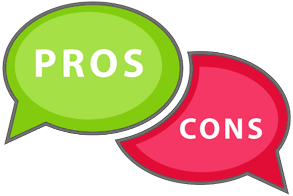
Pros and Cons

Strengths

- local formulation
- uses high order information
- versatile according to the pattern

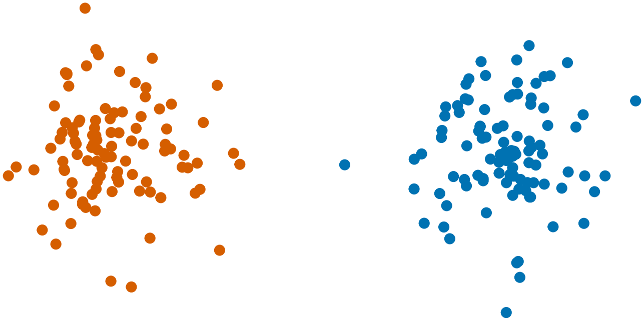
What is missing

- no guarantees
- problem is already hard on a single snapshot
- not very practical computing $P^i, i = 1, \ldots$



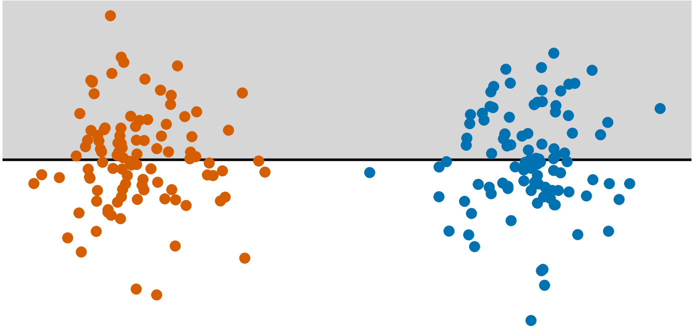
Fairness-Aware Clique Preserving Clustering (Fu et al., 2023)

Fairness definition demographic fairness



Fairness-Aware Clique Preserving Clustering (Fu et al., 2023)

Fairness definition demographic fairness



Fairness-Aware Clique Preserving Clustering (Fu et al., 2023)

Data. $G = \{G_1, \dots, G_T\}$, where V has h different groups

Setting. Find q clusters at each time-point cutting few k-cliques and that are fair.

Problem

Given G parameters k and q, find a q-clustering $C_1^t, \dots, C_q^t$ for $t = 1, \dots, T$ such that

- $\min_{C_i^t} \sum_{t=1}^{T} \sum_{j=1}^{q} \frac{\partial(C_i^t, k)}{vol(C_i^t, k)}$ ($\partial(C_i^t, k)$: $\#$ of k-cliques cut by C_i^t)
- $\frac{|V_S \cap C_i^t|}{|C_i^t|} = \frac{|V_s|}{|V|}$ for each time $t = 1, \dots, T$ and cluster $s = 1, \dots, q$

Fairness-Aware Clique Preserving Clustering (Fu et al., 2023)

Data. $G = \{G_1, \dots, G_T\}$, where V has h different groups

Setting. Find q clusters at each time-point cutting few k-cliques and that are fair.

Problem

Given G parameters k and q, find a q-clustering $C_1^t, \dots, C_q^t$ for $t = 1, \dots, T$ such that

- $\min_{C_i^t} \sum_{t=1}^{T} \sum_{j=1}^{q} \frac{\partial(C_i^t, k)}{vol(C_i^t, k)}$ ($\partial(C_i^t, k)$: # of k-cliques cut by C_i^t)
- $\frac{|V_S \cap C_i^t|}{|C_i^t|} = \frac{|V_s|}{|V|}$ for each time $t = 1, \dots, T$ and cluster $s = 1, \dots, q$

Computing a solution

The problem is **NP**-hard

At fixed t solution is based on spectral techniques: trace minimization problem $+$ K-means.

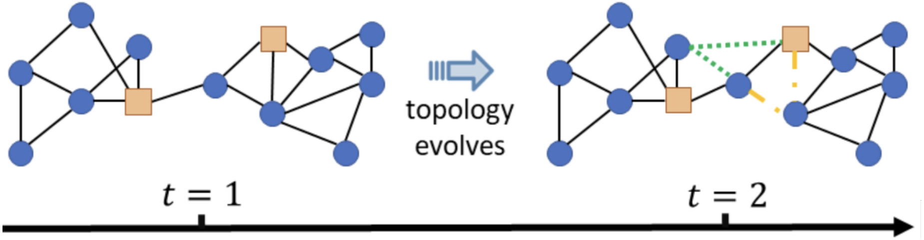
Avoid re computation at each time t, accounting for edge additions and deletions.

Runtime is $\mathcal{O}(T(q^4 + q^2 n) + \sum_t k a^{k-2} m^t)$, a is arboricity at time t, no guarantees on the solution.

Visualizing the Desired Behavior

Example

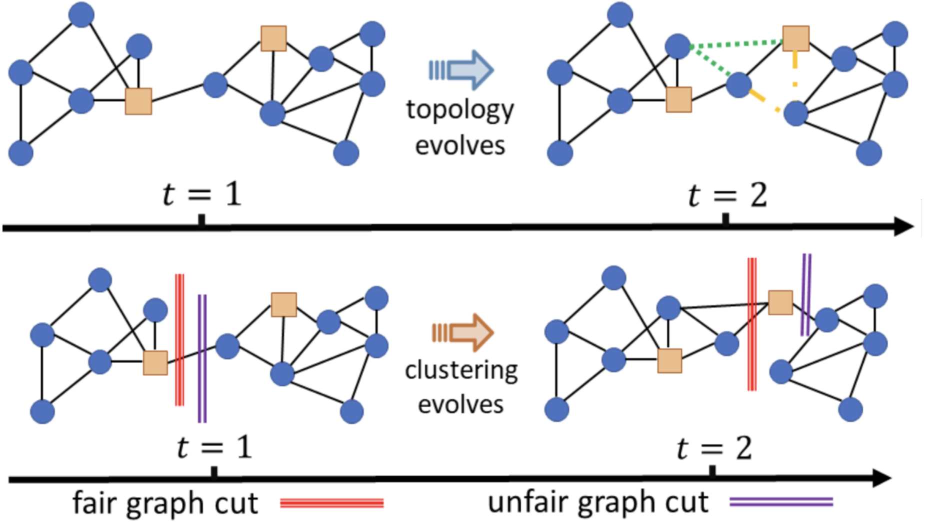
Let $q = 2$ and $k = 3$ (k-clique is a triangle). **Green edges** are insertions and **yellow edges** are removals



Visualizing the Desired Behavior

Example

Let $q = 2$ and $k = 3$ (k-clique is a triangle). **Green edges** are insertions and **yellow edges** are removals



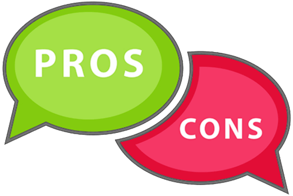
Pros and Cons

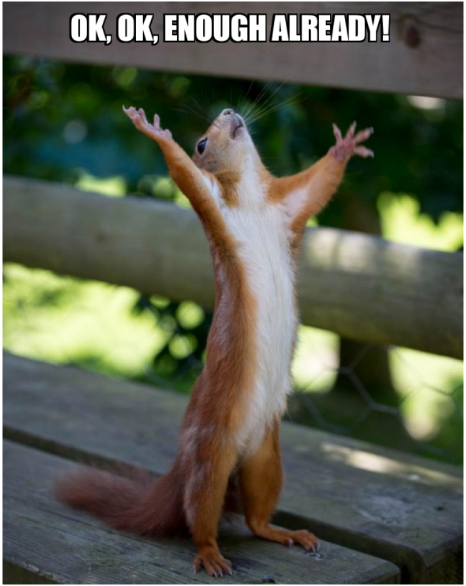
Strengths

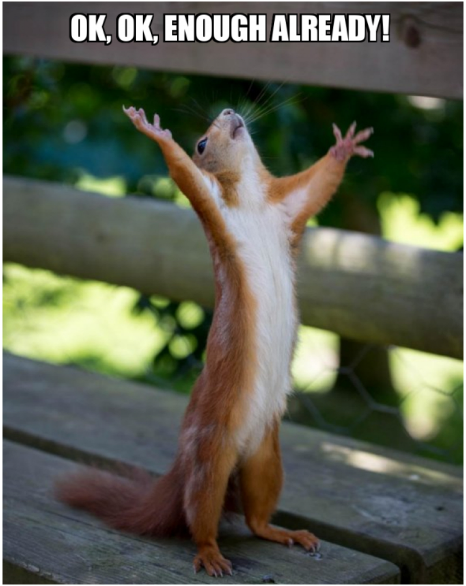
- fairness + evolving networks
- global dense clusters
- can have different applications

What is missing

- no guarantees
- problem is already hard on a single snapshot
- not very practical enumerating k-cliques




OK, OK, ENOUGH ALREADY!



Let us do a summary.

Community Detection in Temporal Networks – Summary

- **community detection in temporal networks** is a very wide research area
- do not panic and follow a **principled approach** (start from data!)
  - identify **properties of the communities** you are looking (global vs. local, etc..)
  - **search formulations** with desired properties (*much work has been done!*)
  - if nothing works you found a **gap in literature** (novel algorithms are needed!)

- **use/develop proper algorithms** to analyze temporal communities

Keep in mind. There is a **gap between formulations and applications**

Temporal communities – Other Formulations

Some other existing formulations

- (Lin et al., 2022) find multiple maximal quasi-clique based communities, stable overall and with interval-based edges
- (Qin et al., 2023) find single and dense community that is periodic over time
- (Preti et al., 2021) discovering a set of diverse and correlated communities in dynamic setting
- (Ma et al., 2020) finding dense subgraphs in temporal networks with time-varying edge weight
- (Banerjee and Pal, 2022) online algorithm for temporal clique identification
- (Chu et al., 2019) bursty and dense community identification
- ...


Coffee time
Right meow
quickmeme.com

Temporal Motifs and Events

Subgraph Motifs

Motifs are small **subgraph patterns** with a plethora of applications in various domains

- databases
- biology
- social networks
- e-commerce

Subgraph Motifs

Motifs are small **subgraph patterns** with a plethora of applications in various domains

- databases
- biology
- social networks
- e-commerce

But how?

Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \Longrightarrow (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

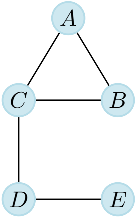
Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

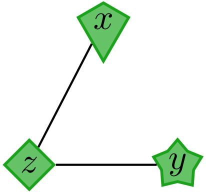
- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \implies (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

Example

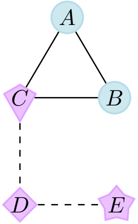


(a) G



(b) H

Is "$V[C, D, E]$" isomorphic to H?



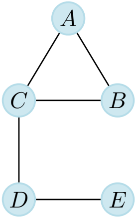
Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

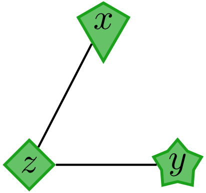
- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \implies (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

Example

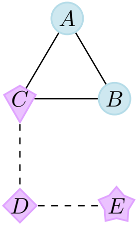


(a) G



(b) H

Is "$V[C, D, E]$" isomorphic to H? **Yes!**



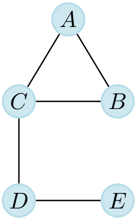
Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

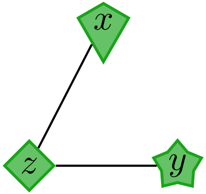
- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \implies (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

Example

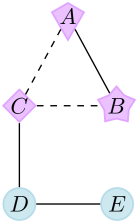


(a) G



(b) H

Is "$V[A, C, B]$" isomorphic to H?



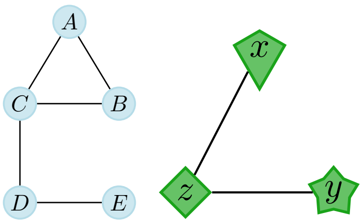
Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

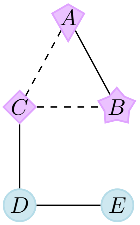
$$(x, y) \in E_H \Longrightarrow (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

Example



(a) G (b) H

Is "$V[A, C, B]$" isomorphic to H? **Yes!**



Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

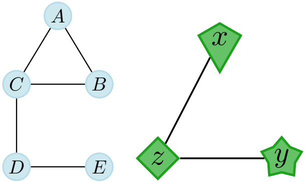
- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \implies (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

- the isomorphism is called *induced* if it also holds

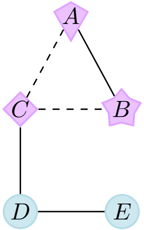
$$(x, y) \in E_H \iff (f(x), f(y)) \in E_{G'}$$

Example



(a) G (b) H

Is "$V[A, C, B]$" isomorphic to H?



Short Primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

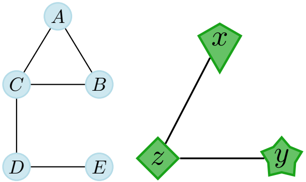
- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \implies (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

- the isomorphism is called *induced* if it also holds

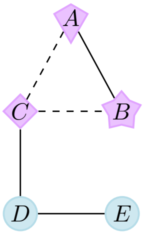
$$(x, y) \in E_H \iff (f(x), f(y)) \in E_{G'}$$

Example



(a) G (b) H

Is "$V[A, C, B]$" isomorphic to H? **No!**



$(f^{-1}(A), f^{-1}(B))$ not in H!

Short primer on Subgraph Isomorphism

Given a simple static graph $G = (V, E)$ and small target graph $H = (V_H, E_H)$ we say that,

- $G' \subseteq G$ is a subgraph that is isomorphic to H if there exists a bijection $f : V_H \mapsto V_{G'}$ with

$$(x, y) \in E_H \Longrightarrow (f(x), f(y)) \in E_{G'} \qquad (G' \sim H)$$

- the isomorphism is called *induced* if it also holds

$$(x, y) \in E_H \Longleftrightarrow (f(x), f(y)) \in E_{G'}$$

Given a graph G, count of H meaning: $\#$ of distinct subgraphs $G' \subseteq G$ with $G' \sim H$.
If $G' \sim H$ we say G' is an *occurrence* of H

Problem

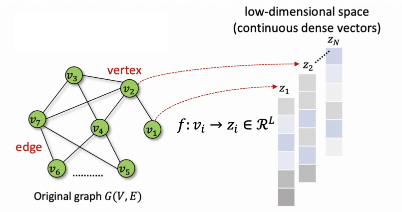
Given a graph G and a *small* subgraph pattern H

- obtain the count of H (counting problem)
- list all occurrences of H

Subgraph Counts

The problem is **NP**-hard and extremely challenging

Many applications, in *computer science*, *network science* and more...

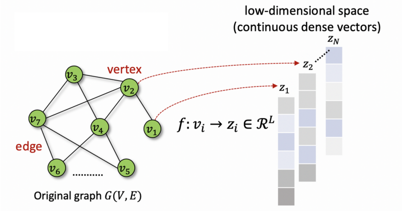


(a) Node embeddings

Subgraph Counts

The problem is **NP**-hard and extremely challenging

Many applications, in *computer science*, *network science* and more...



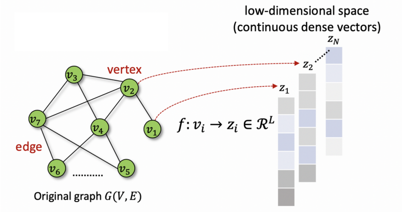
(a) Node embeddings

(b) Malware detection

Subgraph Counts

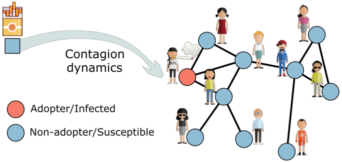
The problem is **NP**-hard and extremely challenging

Many applications, in *computer science*, *network science* and more...



(a) Node embeddings

(b) Malware detection



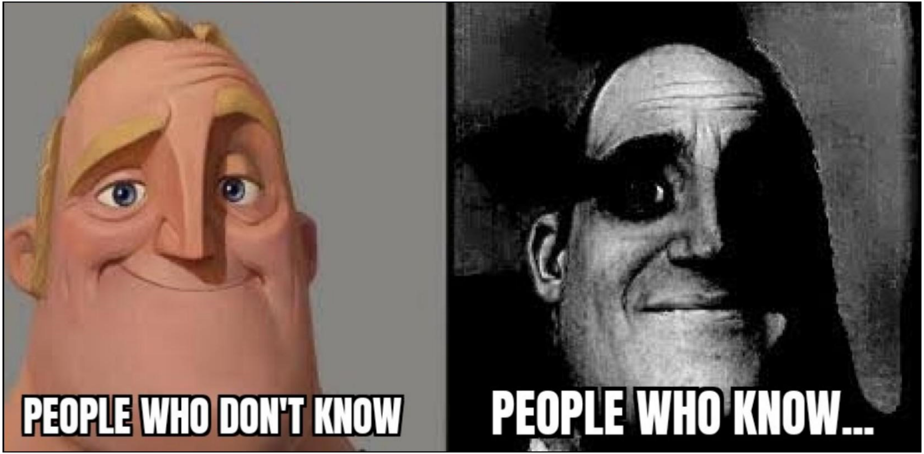
(c) Spreading processes

Some material

- (Ribeiro et al., 2021) survey on algorithms and applications
- (Seshadhri and Tirthapura, 2019) tutorial in WWW 2019 on subgraph counting

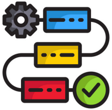
What about temporal motifs?

Temporal Motifs



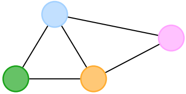
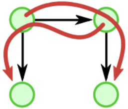
Temporal Motifs

As for temporal communities many definitions exist

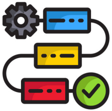


$$\text{Temporal motifs} = \text{static subgraphs} + \text{temporal dynamics} \; (+ \text{additional information})$$

Where

- static subgraph may be (non)induced 
- temporal dynamics time over the static subgraph, in *many ways* 

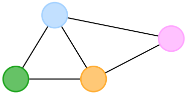
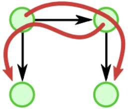
Temporal Motifs



As for temporal communities many definitions exist

Temporal motifs = static subgraphs + temporal dynamics (+ additional information)

Where

- static subgraph may be (non)induced 
- temporal dynamics time over the static subgraph, in *many ways* 
- additional information is any available *metadata* on nodes or edges, e.g., $f : \{V, E\} \to D$

Let us see some of most used definitions

Temporal Motifs by Kovanen et al. (2011)

Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.

Temporal Motifs by Kovanen et al. (2011)

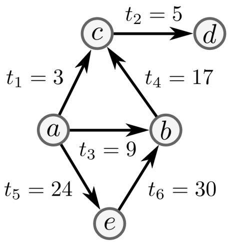
Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

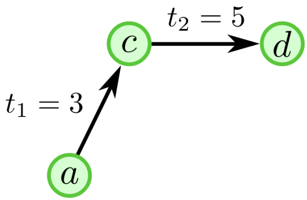
Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.

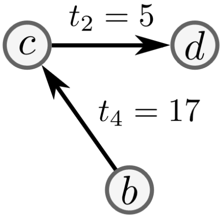
Example. Fix $\Delta t = 10$



(a) Temporal Network G



(b) Δt-adjacent edges



(c) Not Δt-adjacent

Temporal Motifs by Kovanen et al. (2011)

Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.
- two edges ($e_1, e_2 \in E$) are Δt-*connected* if there is a sequence Δt-adj. edges from e_1 to e_2 (no time ordering!)

Temporal Motifs by Kovanen et al. (2011)

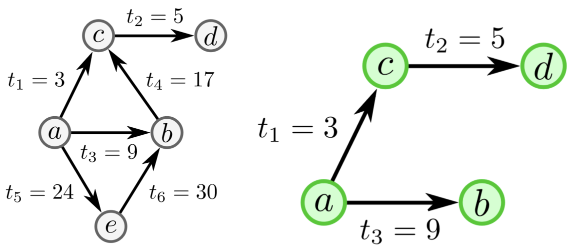
Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.
- two edges ($e_1, e_2 \in E$) are Δt-*connected* if there is a sequence Δt-adj. edges from e_1 to e_2 (no time ordering!)

Example. Fix $\Delta t = 10$



(a) Temporal Network G (b) Δt-connected edges

Temporal Motifs by Kovanen et al. (2011)

Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.
- two edges ($e_1, e_2 \in E$) are Δt-*connected* if there is a sequence Δt-adj. edges from e_1 to e_2 (no time ordering!)

Further, a *connected temporal subgraph* $G' \subseteq G$ constitutes of pairwise Δt-conn. edges

- a conn. temporal subgraph G' is *valid* if for each two events that are incident to a node no other edges are skipped (temporally induced G')

Temporal Motifs by Kovanen et al. (2011)

Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

Modeling temporal dynamics

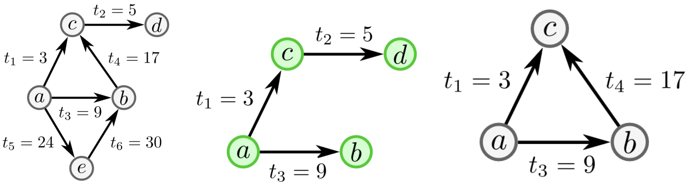
Some definitions

- two edges ($e_1, e_2 \in E$) are Δt-*adjacent* they share at least one node and $|t_{e_1} - t_{e_2}| \leq \Delta t$.
- two edges ($e_1, e_2 \in E$) are Δt-*connected* if there is a sequence Δt-adj. edges from e_1 to e_2 (no time ordering!)

Further, a *connected temporal subgraph* $G' \subseteq G$ constitutes of pairwise Δt-conn. edges

- a conn. temporal subgraph G' is *valid* if for each two events that are incident to a node no other edges are skipped (temporally induced G')

Example. Fix $\Delta t = 10$



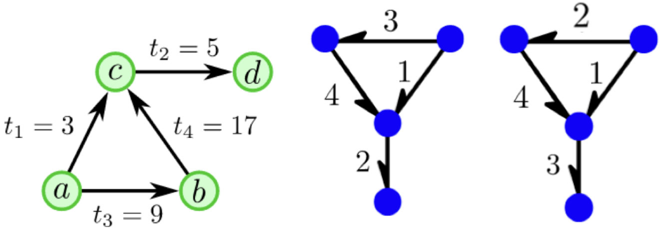
(a) Temporal Network G (b) Valid (c) Not valid

Temporal Motifs by (Kovanen et al., 2011)

Definition

Temporal motifs are *non-isomorphic* classes of subgraphs, where the isomorphism *takes into account edge ordering*.

Example.



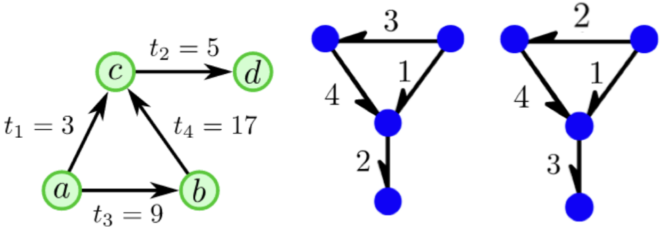
(a) Motif occurrence (b) Class 1? (c) Class 2?

Temporal Motifs by (Kovanen et al., 2011)

Definition

Temporal motifs are *non-isomorphic* classes of subgraphs, where the isomorphism *takes into account edge ordering*.

Example.



(a) Motif occurrence (b) Class 1? ✓ (c) Class 2? ✗

Temporal Motifs by (Kovanen et al., 2011)

Definition

Temporal motifs are *non-isomorphic* classes of subgraphs, where the isomorphism *takes into account edge ordering*.

Problem

Given G, Δt and a bound k obtain the count of temporal motifs on k nodes.

Solving the counting problem for specific classes

- pre-process G and identify maximal components $\mathcal{O}(|E|)$
- in each component find valid subgraphs of bounded size k, $\mathcal{O}(n^k)$
- map each valid subgraph G' with k-edges on its class (*canonical labeling* is used, exponential in $|G'|$).

A real use case?

Homophily in Phone Call Networks (Kovanen et al., 2013)

Data. Record of 6 months data of mobile phone calls (625million calls) and SMS (207 million). In total >6 million users.

Additional metadata. Sex, age and payment type, combined to obtain 24 different *node colors*

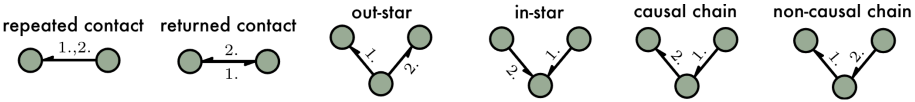
Homophily in Phone Call Networks (Kovanen et al., 2013)

Data. Record of 6 months data of mobile phone calls (625million calls) and SMS (207 million). In total > 6 million users.

Additional metadata. Sex, age and payment type, combined to obtain 24 different *node colors*

Insights. *Temporal homophily* occurs at gender level, i.e., temporal phone call patterns tend to be different in males and females

Temporal motifs.



The score denotes how strong is data with respect to a random-model. F: female, M: male, $F-F$: all nodes are of F class, $F-*$: there exist at least one node of M class in the motif.

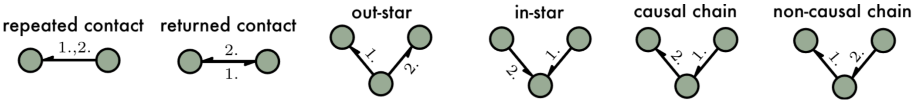
Homophily in Phone Call Networks (Kovanen et al., 2013)

Data. Record of 6 months data of mobile phone calls (625million calls) and SMS (207 million). In total > 6 million users.

Additional metadata. Sex, age and payment type, combined to obtain 24 different *node colors*

Insights. *Temporal homophily* occurs at gender level, i.e., temporal phone call patterns tend to be different in males and females

Temporal motifs.

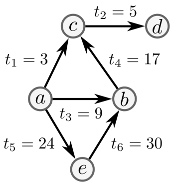


The score denotes how strong is data with respect to a random-model. F: female, M: male, $F-F$: all nodes are of F class, $F-*$: there exist at least one node of M class in the motif.

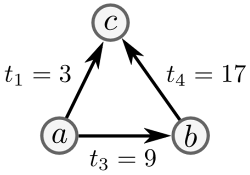
Motif	$F-F$	$F-*$	$M-M$	$M-*$
Repeated calls	1.11	1.11	1.13	1.10
Noncausal chain	**1.08**	**1.02**	1.01	1.04
Causal chain	**1.08**	**1.01**	**0.98**	**1.02**
Out-star	**1.10**	**1.03**	**1.01**	**1.04**

An Issue with Kovanen et al. (2011)'s Definition

Example. Fix $\Delta t = 10$



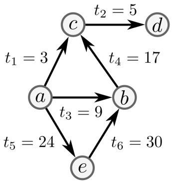
(a) Temporal Network G



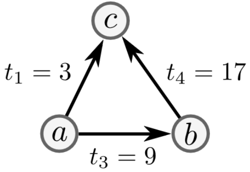
(b) Not valid

An Issue with Kovanen et al. (2011)'s Definition

Example. Fix $\Delta t = 10$



(a) Temporal Network G



(b) Not valid

Considering only *valid* subgraphs may be **too strict**!

Which can lead to *information loss*, as this can be important for many applications.

How to fix this?

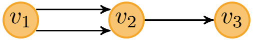
Temporal Motifs by Paranjape et al. (2017)

This model aims at providing a more general and flexible definition of temporal motifs.
Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

A temporal motif is a pair $M = (K, \sigma)$ (Liu et al., 2019) where

- K is a directed and (weakly)connected multigraph with k-nodes and ℓ-edges.

(a) K_1



(b) K_2

Temporal Motifs by Paranjape et al. (2017)

This model aims at providing a more general and flexible definition of temporal motifs.
Data. $G = (V, E), E = \{(u, v, t)\}$ be a *directed* temporal network

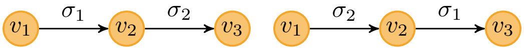
A temporal motif is a pair $M = (K, \sigma)$ (Liu et al., 2019) where

- K is a directed and (weakly)connected multigraph with k-nodes and ℓ-edges.

(a) K_1 (b) K_2

- σ is an ordering of the edges of K (modelling temporal dynamics of K)

  Example. Fixing $K = K_1$ then



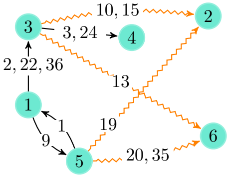
(a) $\sigma_L = \langle (v_1, v_2), (v_2, v_3) \rangle$ (b) $\sigma_R = \langle (v_2, v_3), (v_1, v_2) \rangle$

Note. σ_L is time respecting while σ_R not!

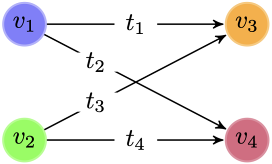
Temporal Motif Counting Problem (Paranjape et al., 2017)

Given G and a value $\delta \in \mathbb{R}^+$, a *time-ordered* sequence $S = \langle (x'_1, y'_1, t'_1), \ldots, (x'_\ell, y'_\ell, t'_\ell) \rangle$ of ℓ unique edges from G is a δ-*instance* of $M = \langle (x_1, y_1), \ldots, (x_\ell, y_\ell) \rangle$ if

1. there exists a bijection h from the vertices appearing in S to the vertices of M, with $h(x'_i) = x_i$ and $h(y'_i) = y_i$, and $i \in [\ell]$;
2. the edges of S occur within δ time; i.e., $t'_\ell - t'_1 \leq \delta$.



(a) Temporal graph G

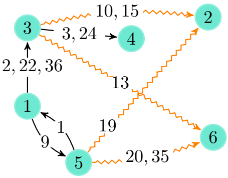


(b) Temporal motif, t_i gives ordering in σ

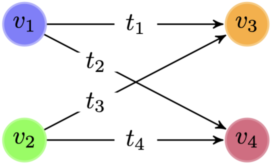
Temporal Motif Counting Problem (Paranjape et al., 2017)

Given G and a value $\delta \in \mathbb{R}^+$, a *time-ordered* sequence $S = \langle (x'_1, y'_1, t'_1), \ldots, (x'_\ell, y'_\ell, t'_\ell) \rangle$ of ℓ unique edges from G is a δ-*instance* of $M = \langle (x_1, y_1), \ldots, (x_\ell, y_\ell) \rangle$ if

1. there exists a bijection h from the vertices appearing in S to the vertices of M, with $h(x'_i) = x_i$ and $h(y'_i) = y_i$, and $i \in [\ell]$;
2. the edges of S occur within δ time; i.e., $t'_\ell - t'_1 \leq \delta$.

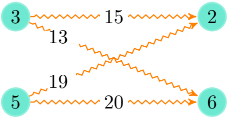


(a) Temporal graph G

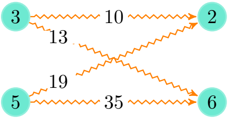


(b) Temporal motif, t_i gives ordering in σ

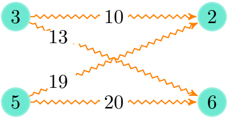
Fix $\delta = 10$



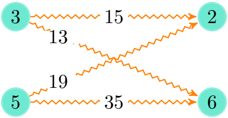
(a) ?



(b) ?



(c) ?

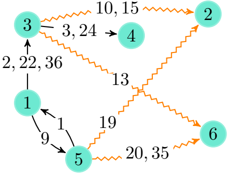


(d) ?

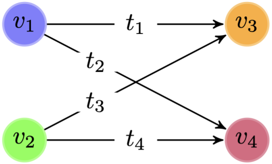
Temporal Motif Counting Problem (Paranjape et al., 2017)

Given G and a value $\delta \in \mathbb{R}^+$, a *time-ordered* sequence $S = \langle (x'_1, y'_1, t'_1), \ldots, (x'_\ell, y'_\ell, t'_\ell) \rangle$ of ℓ unique edges from G is a δ-*instance* of $M = \langle (x_1, y_1), \ldots, (x_\ell, y_\ell) \rangle$ if

1. there exists a bijection h from the vertices appearing in S to the vertices of M, with $h(x'_i) = x_i$ and $h(y'_i) = y_i$, and $i \in [\ell]$;
2. the edges of S occur within δ time; i.e., $t'_\ell - t'_1 \leq \delta$.

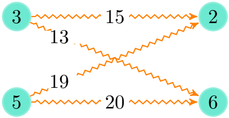


(a) Temporal graph G

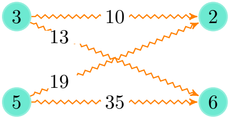


(b) Temporal motif, t_i gives ordering in σ

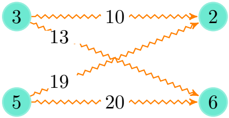
Fix $\delta = 10$



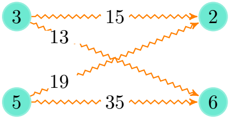
(a) ? ✗



(b) ? ✗



(c) ? ✓



(d) ? ✗

Temporal Motif Counting Problem – cont.

Count of a temporal motif M is: $\#$ of δ-instances of M in G

Problem

Given a temporal network G, a temporal motif M and a parameter $\delta \in \mathbb{R}^+$ obtain *the count* of the temporal motif M

The problem is **NP**-hard

Temporal Motif Counting Problem – cont.

Count of a temporal motif M is: $\#$ of δ-instances of M in G

Problem

Given a temporal network G, a temporal motif M and a parameter $\delta \in \mathbb{R}^+$ obtain *the count* of the temporal motif M

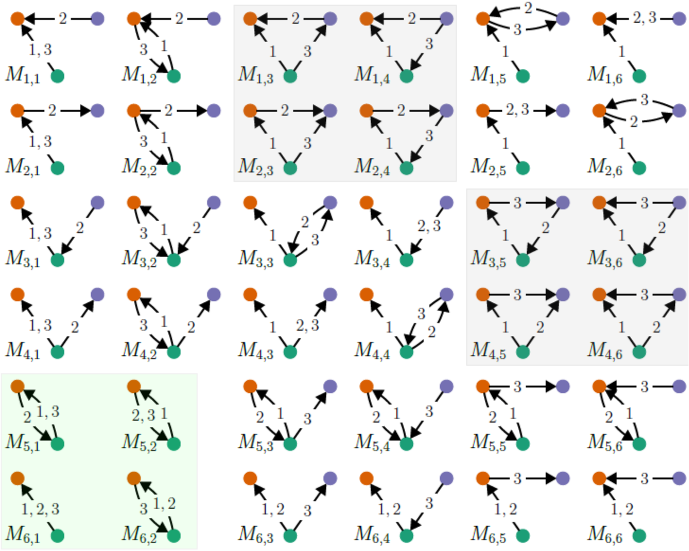
The problem is **NP**-hard

The problem is **NP**-hard even for motifs in **P** for static networks (Liu et al., 2019)!

Lets look at existing algorithms

Exact Algorithm by Paranjape et al. (2017)

The proposed algorithm computes the counts of all $\{2, 3\}$-node 3-edge temporal motifs.



Exact Algorithm by Paranjape et al. (2017)

The proposed algorithm computes the counts of all $\{2,3\}$-node 3-edge temporal motifs.

General framework

- computes the aggregate graph G_A of G
- enumerates all subgraphs $H \subseteq G_A$ isomorphic to K (i.e., $H \sim K$)
- for each H gathers the corresponding temporal networks G_H and sorts edges by timestamps
- applies dynamic-programming to obtain the counts of all the sequences of length ℓ in a window of size δ

Exact Algorithm by Paranjape et al. (2017)

The proposed algorithm computes the counts of all $\{2,3\}$-node 3-edge temporal motifs.

General framework

- computes the aggregate graph G_A of G
- enumerates all subgraphs $H \subseteq G_A$ isomorphic to K (i.e., $H \sim K$)
- for each H gathers the corresponding temporal networks G_H and sorts edges by timestamps
- applies dynamic-programming to obtain the counts of all the sequences of length ℓ in a window of size δ

Runtime. $\mathcal{O}(|E| + n^k + \sum_{H \sim K} |E_{G_H}|(\log(|E_{G_H}|) + |E_H|^{\ell})$

Specialized routines for specific motif classes through dynamic programming

Other Exact Algorithms

Other exact approaches

(Mackey et al., 2018) *enumerates* all δ-instances of a fixed temporal motif M without constrains

(Pashanasangi and Seshadhri, 2021) Fast algorithms for temporal triangle counting based on degeneracy ordering

(Gao et al., 2022) improved algorithms for counting $\{2, 3\}$-node 3-edge temporal motifs

(Sarpe, 2023) improved (Mackey et al., 2018) by different matching criteria and timeline partition

(Yuan et al., 2023) dedicated hardware for counting temporal motifs

(Cai et al., 2023) exact algorithms for counting butterflies in temporal bipartite networks

...

As the problem is hard, often better rely on **approximate counting**!

Motif Approximation Problem

Problem

Given a temporal network G, a temporal motif M and a parameter $\delta \in \mathbb{R}^+$, and two additional parameters $\varepsilon, \eta \in (0,1)^2$ obtain C' an estimate of *the count* C of the temporal motif M with

$$\mathbb{P}[|C' - C| \geq \varepsilon C] \leq \eta$$

Why *approximate counts*?

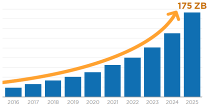
Motif Approximation Problem

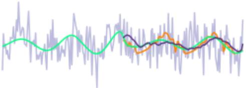
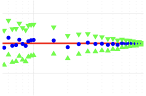
Problem

Given a temporal network G, a temporal motif M and a parameter $\delta \in \mathbb{R}^+$, and two additional parameters $\varepsilon, \eta \in (0,1)^2$ obtain C' an estimate of *the count* C of the temporal motif M with

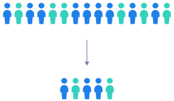
$$\mathbb{P}[|C' - C| \geq \varepsilon C] \leq \eta$$

Why *approximate counts*?

- often efficient and practical to compute on massive data 

- approximations are robust to noisy data 
- guarantees on the quality of estimate 

Approximation Algorithms



Most of approximate algorithm are based on randomized sampling

(Liu et al., 2019): partitioning time-span of G in non-overlapping windows and uses importance sampling to decide windows to explore

(Wang et al., 2020): sampling temporal edges with fixed probability, specialized estimators for triangles, and streaming

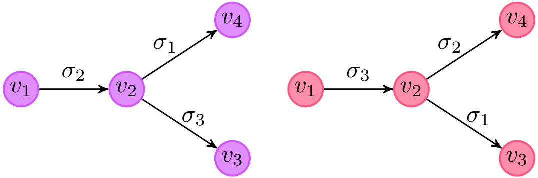
(Sarpe and Vandin, 2021b; Sarpe, 2023): interval based algorithms performing uniform sampling without partitioning

(Pu et al., 2023): edges sampling techniques for counting temporal butterflies on undirected bipartite temporal networks

...

On the Selection of σ (Sarpe and Vandin, 2021a)

A temporal motif is a pair (K, σ), how to properly pick σ?



On the Selection of σ (Sarpe and Vandin, 2021a)

A temporal motif is a pair (K, σ), how to properly pick σ?

Multiple values of σ may need to be tested!

On the Selection of σ (Sarpe and Vandin, 2021a)

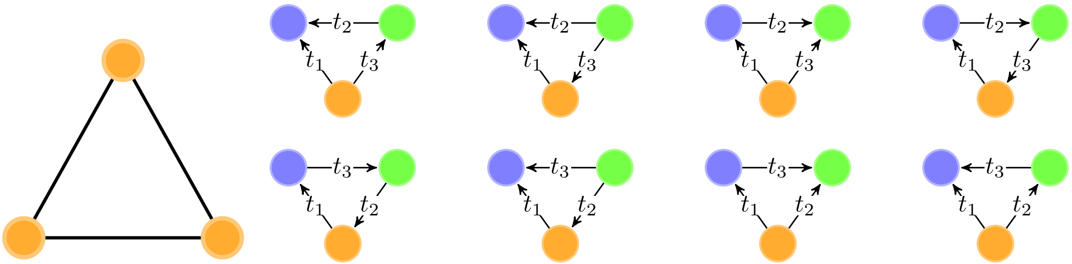
A temporal motif is a pair (K, σ), how to properly pick σ?

Multiple values of σ may need to be tested!

Problem

Given a temporal network G, a parameter $\delta \in \mathbb{R}^+$, a *static* undirected subgraph H and a value $\ell \geq |E_H|$ compute the count of *all* temporal motifs “mapping” on H and having ℓ temporal edges

Example. Fix $\ell = 3$ then

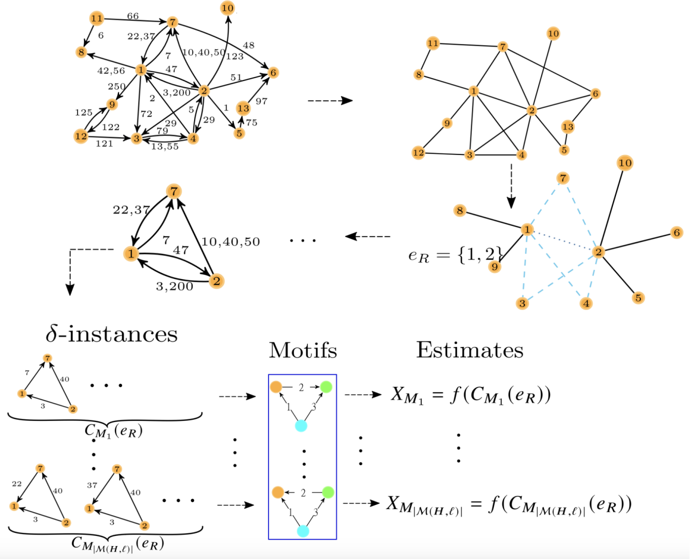


(a) H

(b) motifs to “count”

$\textsc{odeN}$ (Sarpe and Vandin, 2021a)

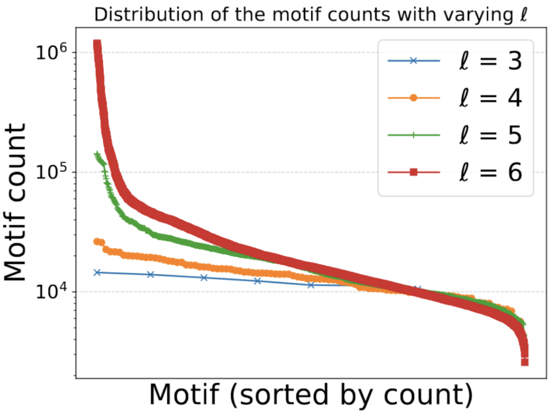
Proposed algorithm $\textsc{odeN}$: randomized sampling algorithm + theoretical guarantees



ODEN (Sarpe and Vandin, 2021a)

Proposed algorithm ODEN: randomized sampling algorithm + theoretical guarantees

Efficiently estimates multiple temporal motifs counts simultaneously. H: triangle and data comes from Facebook posts, varying ℓ.

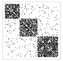


Are there applications?

Are there applications?

Temporal motifs enabled both novel algorithmic problems and more nuanced applications

Stochastic Block Models (Porter et al., 2022)



Goal. Obtain highly accurate stochastic block models (SBM) to capture temporal motif δ-instances

Proposed solution. Temporal Activity SBM

1. partition nodes according to their activity level {in, out}-edges (resp. C^{in}, c^{out} groups)
2. model temporal edges according

$$\theta = \left(\begin{array}{c|c} \theta_{\mathbf{1},\mathbf{1}} & \theta_{\mathbf{1},\mathbf{2}} \\ \hline \theta_{\mathbf{2},\mathbf{1}} & \theta_{\mathbf{2},\mathbf{2}} \end{array}\right)$$

   where $\theta \in \mathbb{R}^{C^{in} \times C^{out}}$ models edge occurrence

3. analytical computation of motif counts according to such model

Stochastic Block Models (Porter et al., 2022)

Goal. Obtain highly accurate stochastic block models (SBM) to capture temporal motif δ-instances

Proposed solution. Temporal Activity SBM

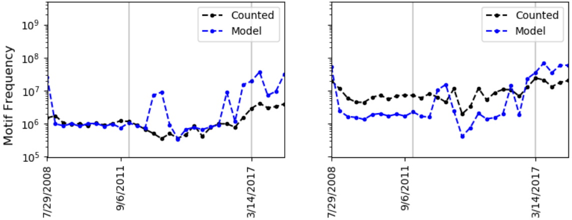
1. partition nodes according to their activity level {in, out}-edges (resp. C^{in}, c^{out} groups)
2. model temporal edges according

$$\theta = \left(\begin{array}{c|c} \boldsymbol{\theta_{1,1}} & \boldsymbol{\theta_{1,2}} \\ \hline \boldsymbol{\theta_{2,1}} & \boldsymbol{\theta_{2,2}} \end{array}\right)$$

   where $\theta \in \mathbb{R}^{C^{in} \times C^{out}}$ models edge occurrence

3. analytical computation of motif counts according to such model

The **model accurately tracks temporal motif counts**. Financial dataset recorded over 10 years, $\delta = 90$ days, left $M_1 = \langle (v_1, v_2), (v_3, v_2), (v_1, v_2) \rangle$, right $M_2 = \langle (v_1, v_2), (v_2, v_1), (v_2, v_1) \rangle$



Synthetic Network Generators (Liu and Sariyüce, 2023)

Goal. Obtain a synthetic temporal network, similar to the one in input for temporal motifs

Proposed solution. Motif Transition Model

Cold event (CE): first event on a temporal motif instance

1. compute temporal network statistics
   - static degree distribution (K_{CE}) and timestamps (T_{CE}) of cold events
   - $\mathcal{P}$ motif-transition properties (how likely are motifs to evolve from one to another)
   - Λ motif transition rated (how often they transition)
   - μ number of static edges involved in transitions.
2. generate static network from K_{CE} and assign (T_{CE})
3. simulate transition process according to computed metrics

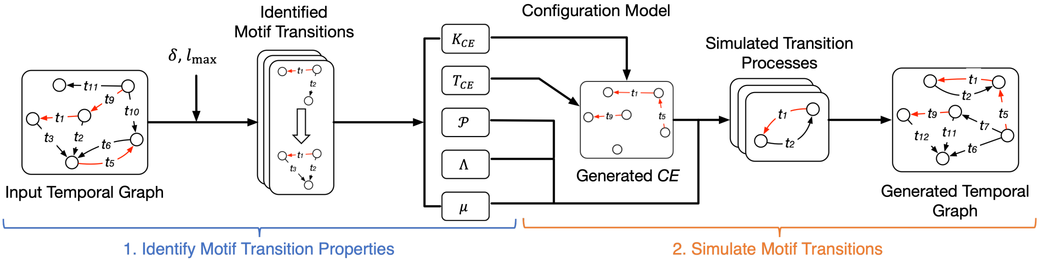
Synthetic Network Generators (Liu and Sariyüce, 2023)

Goal. Obtain a synthetic temporal network, similar to the one in input for temporal motifs

Proposed solution. Motif Transition Model

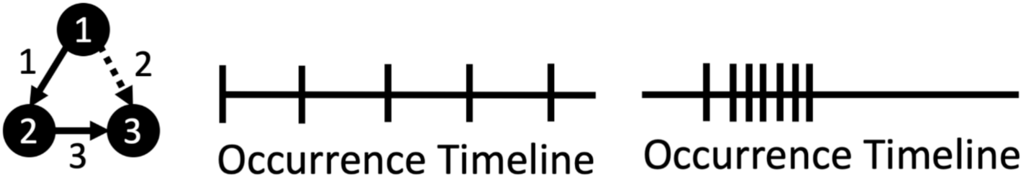
Cold event (CE): first event on a temporal motif instance

1. compute temporal network statistics
   - static degree distribution (K_{CE}) and timestamps (T_{CE}) of cold events
   - $\mathcal{P}$ motif-transition properties (how likely are motifs to evolve from one to another)
   - Λ motif transition rated (how often they transition)
   - μ number of static edges involved in transitions.
2. generate static network from K_{CE} and assign (T_{CE})
3. simulate transition process according to computed metrics



Mining Persistent Events (Belth et al., 2020)

Goal. Distinguish between how motif occur over time *streams*



Mining Persistent Events (Belth et al., 2020)

Goal. Distinguish between how motif occur over time *streams*

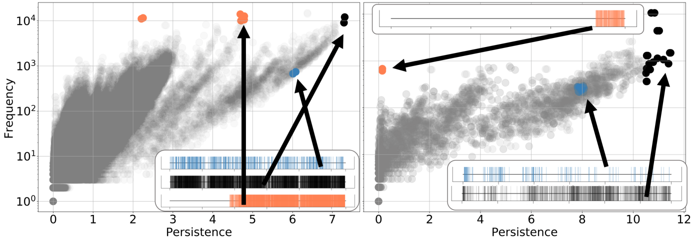
Proposed solution. Assign persistence score and algorithms to compute it

Let x be an event (e.g., temporal motif instance) then the persistence $P(\cdot)$

$$P(x) = f\left(\underbrace{W(x)}_{\text{width}}, \underbrace{F(x)}_{\text{frequency}}, \underbrace{S(x)}_{\text{uniformity}}\right)$$

Online and offline streaming algorithms are developed, efficient for small size of events (small ℓ)

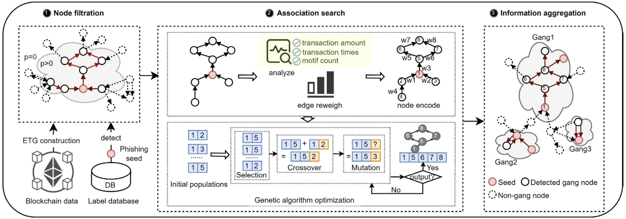
The score allows to distinguish between frequent/infrequence and bursty/persistent

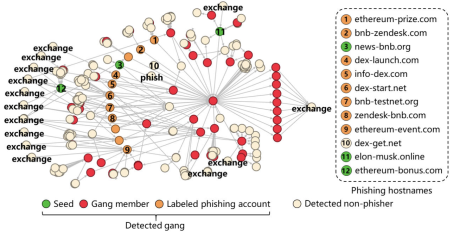


(a) Boston Bike **(b)** Columbus Bike

Some Practical Applications (Liu et al., 2024)

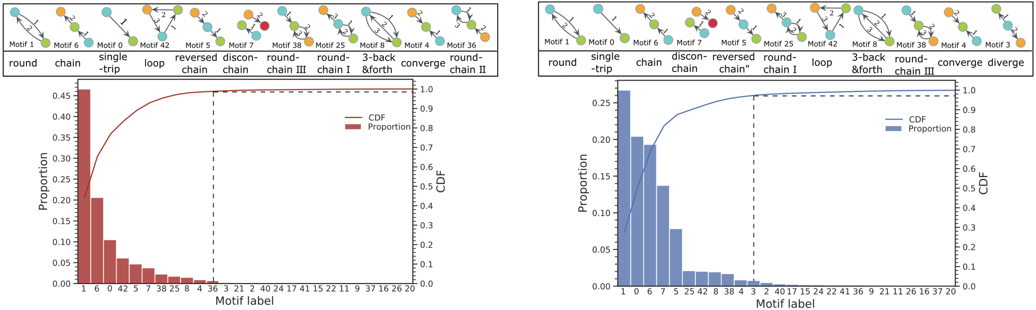
- capturing high-order patterns for phishing gang identification on cryptocurrency networks





Some Practical Applications (Lei et al., 2020)

- analyzing different temporal travel patterns in people commuting (metro vs bike sharing)

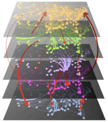


(a) using metro (b) using bikesharing

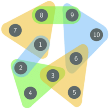
Other Temporal Motif Definitions

Several other definitions exist in literature

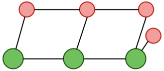
- (Boekhout et al., 2019): studied temporal multilayer motifs



- (Lee and Shin, 2023): studied temporal hypergraph motifs

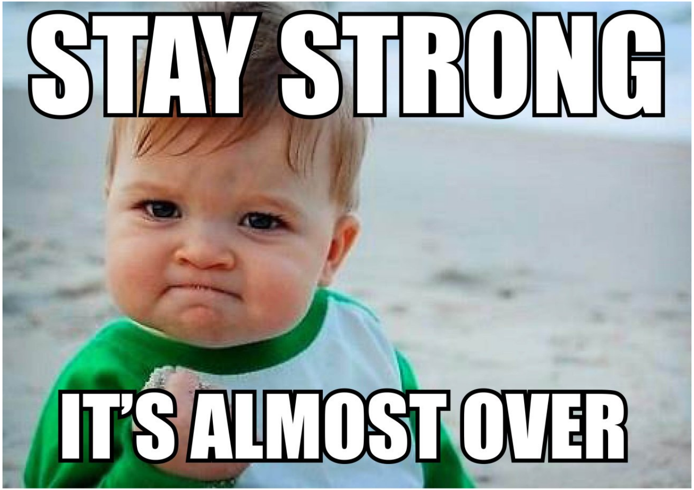


- (Longa et al., 2021): studied motifs based ego-networks



- (Kosyfaki et al., 2018): defined motifs for temporal networks with flows
- ...

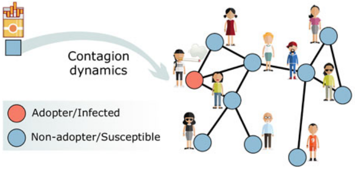
If you want to know more, check the survey by Liu et al. (2021)


STAY STRONG
IT'S ALMOST OVER

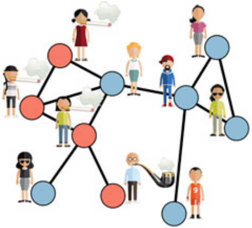
Diffusion and Random Networks

Diffusion Analysis and Spreading

- propagation models
  - used to study disease spreading or information cascade in the network
- activity spreading: virus, information, idea, rumor
- applications: epidemiology, information security, marketing
- why use models?
  - facilitate mathematical analysis of propagation processes
  - have intuitive interpretation
  - proven to be realistic by empirical studies
- extensive survey in the book (Shakarian et al., 2015)



(a) $t = 1$



(b) $t = 2$

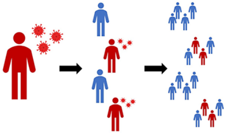
Standard Models

Most used models are

- susceptible-infected (SI) model
  - SIR, SIRS, other variants
- independent cascade (IC) model
- linear threshold (LT) model

Such models are *important building blocks* for many data mining formulations!

Susceptible-Infectious (SI) Model

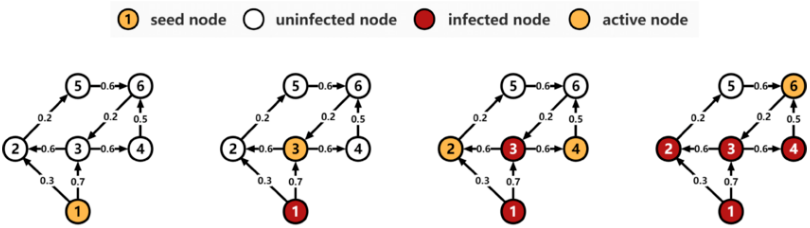


- beginning
  - time step t_0
  - one or several infected nodes in I_{t_0} (seeds of infection)
- subsequent timestamp t
  - all infected nodes try to infect each of their *susceptible* neighbors
  - with probability p infection is *passed through an edge*
  - if a node receives infection becomes infected
- the process continues until all nodes are infected.

Some *other node types*, recovered (nodes that were infectious and now cannot spread), and exposed (infected that cannot spread)

Independent Cascade (IC) Model

- nodes can be in either susceptible or infectious
- each edge (u, v) has an *individual* infection probability (based on proximity, frequency, etc..)
- infected node u has a *single* chance to infect its neighbors



(a) Time step 0 (b) Time step 1 (c) Time step 2 (d) Time step 3

Used to study new propagation of ideas, concepts, or products (Kempe et al., 2003; Wang et al., 2012)

Linear Threshold (LT) Model

- every edge (u, v) has a probability $p(u, v)$
- at the time step t, u: susceptible $\to$ infectious, if the total weight from its infectious neighbors is larger than a random propagation threshold θ_u

$$\sum_{v \in N(u)} p(v, u)\mathbf{1}[v \text{ is infectious at } t] \geq \theta_u$$

- conditional on thresholds and the initially infected nodes the process is *deterministic*.

LT model has applications in viral-marketing (Chen et al., 2010; Goyal et al., 2011)

Mining Applications

Powerful modeling for many **mining primitives**

1. **immunization strategies**, e.g., find smallest set of nodes to stop a spreading process. (Lee et al., 2012; Yu et al., 2010; Starnini et al., 2013; Génois et al., 2015; Mantzaris and Higham, 2016; Valdano et al., 2015; Gauvin et al., 2015)
2. **influence maximization**, e.g., select the initial set of seeds, to optimize diffusion, applications in marketing and network design. (Aggarwal et al., 2012; Zhuang et al., 2013; Gayraud et al., 2015; Rodriguez et al., 2011; Gomez-Rodriguez et al., 2016; Chen et al., 2012; Liu et al., 2012; Rodriguez and Schölkopf, 2012; Du et al., 2013)
3. **seed and cascade reconstruction**, e.g., given some observed data of a spreading phenomenon, find the *most probable seed nodes* or *cascades*, applications in epidemiology and nfluencer discovery. (Shah and Zaman, 2011; Lappas et al., 2010; Prakash et al., 2012)

Random Models

Common questions in temporal data analysis

- *how novel* is this result?
- is this only due to *random chance*?
- are there *properties in the data explaining* the results?

To find an **answer** $\rightarrow$ use a **statistical test** (Pellegrina et al., 2019)

Random Models

Common questions in temporal data analysis

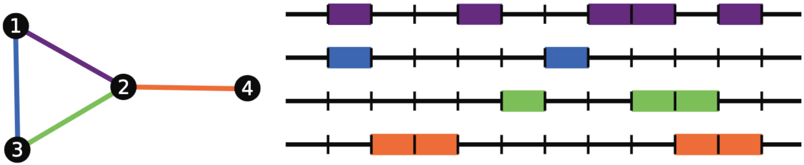
- *how novel* is this result?
- is this only due to *random chance*?
- are there *properties in the data explaining* the results?

To find an **answer** $\rightarrow$ use a **statistical test** (Pellegrina et al., 2019)

- start from a **temporal network** G
- formulate an **hypothesis** (H_0) about data (e.g., time does not matter for $f(G)$)
- perform a **test to reject** H_0, usually
  - generate multiple datasets $G_1^{H_0}, \ldots, G_L^{H_0}$ for some large L according to H_0
  - compute some function $g(G_1^{H_0}, \ldots, G_L^{H_0})$ to reject H_0 (e.g., $g(\cdot)$ not explains $f(G)$).

Random Models Gauvin et al. (2022)

Randomized models are used to test temporal/static properties in data
A temporal network as time-line of events



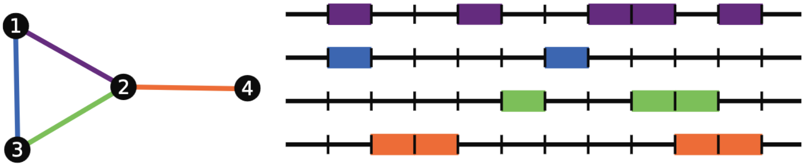
We have

1. static structure (SS)
2. timeline associated to its links (TL)

To obtain random models → use *these two properties or combinations* of the two

Random Models Gauvin et al. (2022)

Randomized models are used to test temporal/static properties in data
A temporal network as time-line of events



We have

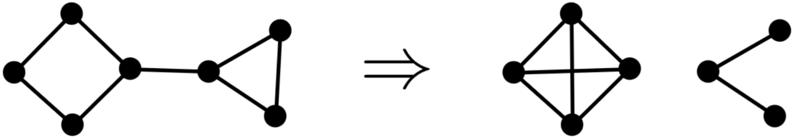
1. static structure (SS)
2. timeline associated to its links (TL)

To obtain random models $\rightarrow$ use *these two properties or combinations* of the two

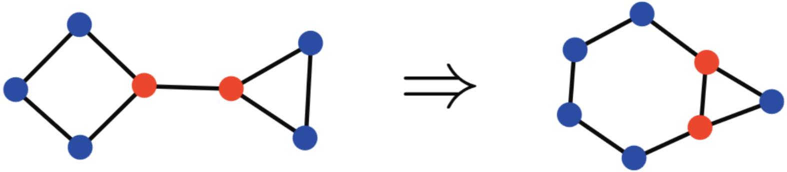
Let us see some examples

Random Models Gauvin et al. (2022)

Shuffling *only* static properties while fixing the temporal ones



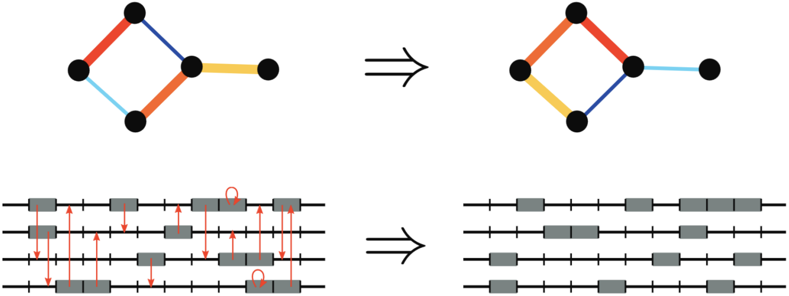
(a) *link shuffling*



(b) *(conn.) constrained degree link shuffling*

Random Models Gauvin et al. (2022)

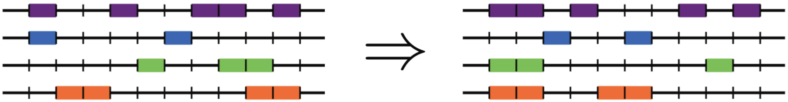
timeline shuffling



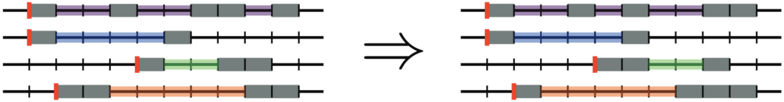
This model retains static properties and conditions on the observed temporal ones

Random Models Gauvin et al. (2022)

Other models retaining static properties



(a) *shuffling events over each timeline*

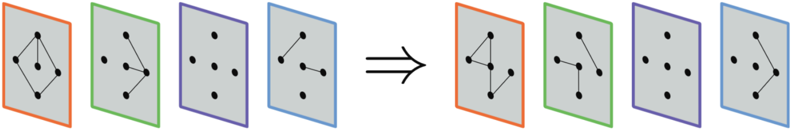


(b) *shuffling events and retaining gaps*

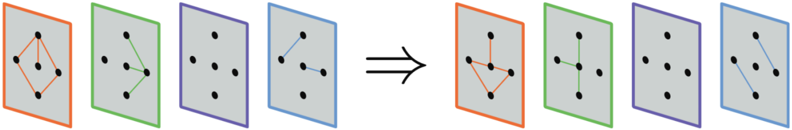
... and much more such combinations (static + temporal) ...

Random Models Gauvin et al. (2022)

Some random models for snapshot-based temporal networks



(a) *snapshot shuffling*



(b) *isomorphism based*

Random Models

Summary.

- *random models* can be of fundamental importance for testing significance/generating additional data
- they can be applied for most of the mining problems that we discussed
- some of them may be *hard to compute* and **new methods may be required**

Agenda

Part I : **Introduction and Motivation**

- models of temporal networks
- algorithmic approaches

Part II : **Mining Temporal Networks A:**

- connectivity, temporal properties
- centrality, cores

Part III : **Mining Temporal Networks B:**

- communities, patterns and events
- diffusion and random networks

Part IV : **Tools and Code Libraries**

Part V : **Challenges, Open Problems, and Trends**

Part IV

Tools and Code Libraries

Tools Overview

Tool Name	Language	Functionalities
SNAP	C++/Python	Temporal motifs
Graph-tool	Python/C++	Simulate network dynamics (e.g., spreading)
Teneto	Python	Temporal network measures (centrality, reachability, etc..), community detection, visualization
Phasik	Python	Infer temporal networks from time series data
Reticula	C++/Python	Random networks, random models, temporal reachability, events...
Tglib	C++/Python	Paths, centrality and other properties (cores, clustering coefficient, etc..)
Raphtory	Rust/Python	Centrality, communities, cores, motifs, null models, visualization and more...

Agenda

Part I : **Introduction and Motivation**

- models of temporal networks
- algorithmic approaches

Part II : **Mining Temporal Networks A:**

- connectivity, temporal properties
- centrality, cores

Part III : **Mining Temporal Networks B:**

- communities, patterns and events
- diffusion and random networks

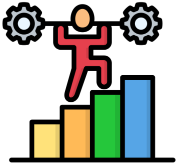
Part IV : **Tools and Code Libraries**

Part V : **Challenges, Open Problems, and Trends**

Part V

Challenges, Open Problems, and Trends

Challenges in Temporal Network Mining



- large number of **problem formulations** and variants
- gaps **fundamental theoretical** treatment
  - many are combinations of several ideas of static cases
  - require often many parameters
- hard to compare methods and **choose based on applications**
  - few datasets with ground-truth solutions
  - synthetic generators are built on various assumptions
  - no standards and benchmarks
  - as always: lack of useful and rich datasets
- a large number of quality metrics to calculate and compare
- comparisons are misleading if methods are designed for other definitions

Directions in Temporal Network Mining

- **more systematic** approaches, **quality guarantees**
- **interpretability** of the results
- **diversity** and **fairness**

- applications and application-tailored algorithms
  - encourage *interdisciplinary research* and collaborations
  - computational *social science*

Thanks for your attention!

https://miningtemporalnetworks.github.io/

menti.com

code: 14 46 97 6

references I

Aggarwal, C. and Subbian, K. (2014). Evolutionary network analysis: A survey. *ACM Computing Surveys (CSUR)*, 47(1):10.

Aggarwal, C. C., Lin, S., and Yu, P. S. (2012). On influential node discovery in dynamic social networks. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 636–647. SIAM.

Akrida, E. C., Gasieniec, L., Mertzios, G. B., and Spirakis, P. G. (2017). The complexity of optimal design of temporally connected graphs. *Theory of Computing Systems*, 61:907–944.

Akrida, E. C., Mertzios, G. B., Spirakis, P. G., and Zamaraev, V. (2020). Temporal vertex cover with a sliding time window. *Journal of Computer and System Sciences*, 107:108–123.

Aynaud, T., Fleury, E., Guillaume, J., Wang, Q., Ganguly, N., Mukherjee, A., Mitra, B., Peruani, F., and Choudhury, M. (2013). Dynamics on and of complex networks.

Banerjee, S. and Pal, B. (2022). An efficient updation approach for enumerating maximal (Δ, γ)-cliques of a temporal network. *Journal of Complex Networks*, 10(5).

Becker, R., Crescenzi, P., Cruciani, A., and Kodric, B. (2023). Proxying betweenness centrality rankings in temporal networks. In *21st International Symposium on Experimental Algorithms (SEA 2023)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.

Belth, C., Zheng, X., and Koutra, D. (2020). Mining persistent activity in continually evolving networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20. ACM.

Berman, K. A. (1996). Vulnerability of scheduled networks and a generalization of menger's theorem. *Networks: An International Journal*, 28(3):125–134.

Bhadra, S. and Ferreira, A. (2003). Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In *Ad-Hoc, Mobile, and Wireless Networks: Second International Conference, ADHOC-NOW2003, Montreal, Canada, October 8-10, 2003. Proceedings 2*, pages 259–270. Springer.

references II

Bi, J., Jin, J., Qu, C., Zhan, X., Wang, G., and Yan, G. (2021). Temporal gravity model for important node identification in temporal networks. *Chaos, Solitons & Fractals*, 147:110934.

Boekhout, H. D., Kosters, W. A., and Takes, F. W. (2019). Efficiently counting complex multilayer temporal motifs in large-scale networks. *Computational Social Networks*, 6(1).

Bonchi, F., Bordino, I., Gullo, F., and Stilo, G. (2019). The importance of unexpectedness: Discovering buzzing stories in anomalous temporal graphs. *Web Intelligence*, 17(3):177–198.

Borgatti, S. P. and Everett, M. G. (2006). A graph-theoretic perspective on centrality. *Social networks*, 28(4):466–484.

Buß, S., Molter, H., Niedermeier, R., and Rymar, M. (2020). Algorithmic aspects of temporal betweenness. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2084–2092.

Cai, X., Ke, X., Wang, K., Chen, L., Zhang, T., Liu, Q., and Gao, Y. (2023). Efficient temporal butterfly counting and enumeration on temporal bipartite graphs. *Proceedings of the VLDB Endowment*, 17(4):657–670.

Casteigts, A., Flocchini, P., Quattrociocchi, W., and Santoro, N. (2012). Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408.

Casteigts, A., Himmel, A.-S., Molter, H., and Zschoche, P. (2021). Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802.

Cencetti, G., Battiston, F., Lepri, B., and Karsai, M. (2021). Temporal properties of higher-order interactions in social networks. *Scientific reports*, 11(1):7028.

Chaintreau, A., Mtibaa, A., Massoulie, L., and Diot, C. (2007). The diameter of opportunistic mobile networks. In *Proceedings of the 2007 ACM CoNEXT conference*, page 12. ACM.

Charikar, M. (2000). Greedy approximation algorithms for finding dense components in a graph. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 84–95. Springer.

references III

Charikar, M., Naamad, Y., and Wu, J. (2018). On finding dense common subgraphs.

Chen, W., Lu, W., and Zhang, N. (2012). Time-critical influence maximization in social networks with time-delayed diffusion process. In *AAAI*, volume 2012, pages 1–5.

Chen, W., Yuan, Y., and Zhang, L. (2010). Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE.

Christopoulos, K. and Tsichlas, K. (2022). *State-of-the-Art in Community Detection in Temporal Networks*, pages 370–381. Springer International Publishing.

Chu, L., Zhang, Y., Yang, Y., Wang, L., and Pei, J. (2019). Online density bursting subgraph detection from temporal graphs. *Proceedings of the VLDB Endowment*, 12(13):2353–2365.

Ciaperoni, M., Galimberti, E., Bonchi, F., Cattuto, C., Gullo, F., and Barrat, A. (2020). Relevance of temporal cores for epidemic spread in temporal networks. *Scientific reports*, 10(1):12529.

Clauset, A. and Eagle, N. (2012). Persistence and periodicity in a dynamic proximity network. *arXiv preprint arXiv:1211.7343*.

Crescenzi, P., Magnien, C., and Marino, A. (2020). Finding top-k nodes for temporal closeness in large temporal graphs. *Algorithms*, 13(9):211.

Cruciani, A. (2023). On approximating the temporal betweenness centrality through sampling. *arXiv preprint arXiv:2304.08356*.

Dakiche, N., Tayeb, F. B.-S., Slimani, Y., and Benatchba, K. (2019). Tracking community evolution in social networks: A survey. *Information Processing & Management*, 56(3):1084–1102.

Du, N., Song, L., Rodriguez, M. G., and Zha, H. (2013). Scalable influence estimation in continuous-time diffusion networks. In *Advances in neural information processing systems*, pages 3147–3155.

Elmezain, M., Othman, E. A., and Ibrahim, H. M. (2021). Temporal degree-degree and closeness-closeness: A new centrality metrics for social network analysis. *Mathematics*, 9(22):2850.

Enright, J., Meeks, K., Mertzios, G. B., and Zamaraev, V. (2021). Deleting edges to restrict the size of an epidemic in temporal networks. *Journal of Computer and System Sciences*, 119:60–77.

Enugala, R., Rajamani, L., Ali, K., and Kurapati, S. (2015). Community detection in dynamic social networks: a survey. *International Journal of Research and Applications*, 2(6):278–285.

Fortunato, S. and Hric, D. (2016). Community detection in networks: A user guide. *Physics reports*, 659:1–44.

Fu, D., Zhou, D., and He, J. (2020). Local motif clustering on time-evolving graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20. ACM.

Fu, D., Zhou, D., Maciejewski, R., Croitoru, A., Boyd, M., and He, J. (2023). Fairness-aware clique-preserving spectral clustering of temporal graphs. In *Proceedings of the ACM Web Conference 2023*, WWW '23. ACM.

Galimberti, E., Ciaperoni, M., Barrat, A., Bonchi, F., Cattuto, C., and Gullo, F. (2020). Span-core decomposition for temporal networks: Algorithms and applications. *TKDD*, 15(1):1–44.

Gao, Z., Cheng, C., Yu, Y., Cao, L., Huang, C., and Dong, J. (2022). Scalable motif counting for large-scale temporal graphs. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.

Gao, Z., Shi, Y., and Chen, S. (2015). Measures of node centrality in mobile social networks. *International Journal of Modern Physics C*, 26(09):1550107.

Gauvin, L., Génois, M., Karsai, M., Kivelä, M., Takaguchi, T., Valdano, E., and Vestergaard, C. L. (2022). Randomized reference models for temporal networks. *SIAM Review*, 64(4):763–830.

Gauvin, L., Panisson, A., Barrat, A., and Cattuto, C. (2015). Revealing latent factors of temporal networks for mesoscale intervention in epidemic spread. *arXiv preprint arXiv:1501.02758*.

references V

Gayraud, N. T., Pitoura, E., and Tsaparas, P. (2015). Diffusion maximization in evolving social networks. In *Proceedings of the 2015 ACM on Conference on Online Social Networks*, pages 125–135. ACM.

Génois, M., Vestergaard, C. L., Fournet, J., Panisson, A., Bonmarin, I., and Barrat, A. (2015). Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers. *Network Science*, 3(3):326–347.

Goh, K.-I. and Barabási, A.-L. (2008). Burstiness and memory in complex systems. *Europhysics Letters*, 81(4):48002.

Goldberg, A. V. (1984). Finding a maximum density subgraph. Technical report, University of California at Berkeley.

Gomez-Rodriguez, M., Song, L., Du, N., Zha, H., and Schölkopf, B. (2016). Influence estimation and maximization in continuous-time diffusion networks. *ACM Transactions on Information Systems (TOIS)*, 34(2):9.

Goyal, A., Lu, W., and Lakshmanan, L. V. (2011). Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 211–220. IEEE.

Gupta, S. and Bedathur, S. (2022). A survey on temporal graph representation learning and generative modeling. *arXiv preprint arXiv:2208.12126*.

Hamm, T., Klobas, N., Mertzios, G. B., and Spirakis, P. G. (2022). The complexity of temporal vertex cover in small-degree graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10193–10201.

Hanauer, K., Henzinger, M., and Schulz, C. (2021). Recent advances in fully dynamic graph algorithms. *arXiv preprint arXiv:2102.11169*.

Hartmann, T., Kappes, A., and Wagner, D. (2016). Clustering evolving networks. In *Algorithm Engineering*, pages 280–329. Springer.

Henzinger, M. R., King, V., and King, V. (1999). Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM (JACM)*, 46(4):502–516.

Himmel, A.-S., Bentert, M., Nichterlein, A., and Niedermeier, R. (2019). Efficient computation of optimal temporal walks under waiting-time constraints. In *International Conference on Complex Networks and Their Applications*, pages 494–506. Springer.

Himmel, A.-S., Molter, H., Niedermeier, R., and Sorge, M. (2017). Adapting the bron–kerbosch algorithm for enumerating maximal cliques in temporal graphs. *Social Network Analysis and Mining*, 7:1–16.

Holme, P. (2015). Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234.

Hu, W., Zou, H., and Gong, Z. (2015). Temporal pagerank on social networks. In *Web Information Systems Engineering–WISE 2015: 16th International Conference, Miami, FL, USA, November 1-3, 2015, Proceedings, Part I 16*, pages 262–276. Springer.

Hung, W.-C. and Tseng, C.-Y. (2021). Maximum (l, k)-lasting cores in temporal social networks. In *DASFAA Intl Workshops*, pages 336–352, Cham.

Kazemi, S. M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., and Poupart, P. (2020). Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73.

Kempe, D., Kleinberg, J., and Tardos, É. (2003). Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM.

Kim, H. and Anderson, R. (2012). Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107.

Klobas, N., Mertzios, G. B., Molter, H., Niedermeier, R., and Zschoche, P. (2023). Interference-free walks in time: Temporally disjoint paths. *Autonomous Agents and Multi-Agent Systems*, 37(1):1.

Klobas, N., Mertzios, G. B., Molter, H., and Spirakis, P. G. (2022). The complexity of computing optimum labelings for temporal connectivity. *arXiv preprint arXiv:2202.05880*.

references VII

Kong, Y.-X., Shi, G.-Y., Wu, R.-J., and Zhang, Y.-C. (2019). k-core: Theories and applications. *Physics Reports*, 832:1–32.

Kosyfaki, C., Mamoulis, N., Pitoura, E., and Tsaparas, P. (2018). Flow motifs in interaction networks.

Kovanen, L., Karsai, M., Kaski, K., Kertész, J., and Saramäki, J. (2011). Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005.

Kovanen, L., Kaski, K., Kertész, J., and Saramäki, J. (2013). Temporal motifs reveal homophily, gender-specific patterns, and group talk in call sequences. *Proceedings of the National Academy of Sciences*, 110(45):18070–18075.

Kujala, R., Weckström, C., Mladenović, M. N., and Saramäki, J. (2018). Travel times and transfers in public transport: Comprehensive accessibility analysis based on pareto-optimal journeys. *Computers, Environment and Urban Systems*, 67:41–54.

Lappas, T., Terzi, E., Gunopulos, D., and Mannila, H. (2010). Finding effectors in social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1059–1068. ACM.

Latapy, M., Viard, T., and Magnien, C. (2018). Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, 8(1):61.

Lee, G. and Shin, K. (2023). Temporal hypergraph motifs. *Knowledge and Information Systems*, 65(4):1549–1586.

Lee, S., Rocha, L. E., Liljeros, F., and Holme, P. (2012). Exploiting temporal network structures of human interaction to effectively immunize populations. *PloS one*, 7(5):e36439.

Lehmann, S. (2019). Fundamental structures in dynamic communication networks. *arXiv preprint arXiv:1907.09966*.

Lei, D., Chen, X., Cheng, L., Zhang, L., Ukkusuri, S. V., and Witlox, F. (2020). Inferring temporal motifs for travel pattern analysis using large scale smart card data. *Transportation Research Part C: Emerging Technologies*, 120:102810.

Li, R.-H., Su, J., Qin, L., Yu, J. X., and Dai, Q. (2018). Persistent community search in temporal networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 797–808. IEEE.

Lin, L., Yuan, P., Li, R.-H., Wang, J., Liu, L., and Jin, H. (2022). Mining stable quasi-cliques on temporal networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(6):3731–3745.

Liu, B., Cong, G., Xu, D., and Zeng, Y. (2012). Time constrained influence maximization in social networks. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 439–448. IEEE.

Liu, J., Chen, J., Wu, J., Wu, Z., Fang, J., and Zheng, Z. (2024). Fishing for fraudsters: Uncovering ethereum phishing gangs with blockchain data. *IEEE Transactions on Information Forensics and Security*, 19:3038–3050.

Liu, P., Benson, A. R., and Charikar, M. (2019). Sampling methods for counting temporal motifs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19. ACM.

Liu, P., Guarrasi, V., and Sariyuce, A. E. (2021). Temporal network motifs: Models, limitations, evaluation. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

Liu, P. and Sariyüce, A. E. (2023). Using motif transitions for temporal graph generation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23. ACM.

Longa, A., Cencetti, G., Lepri, B., and Passerini, A. (2021). An efficient procedure for mining egocentric temporal motifs. *Data Mining and Knowledge Discovery*, 36(1):355–378.

Longa, A., Lachi, V., Santin, G., Bianchini, M., Lepri, B., Lio, P., Scarselli, F., and Passerini, A. (2023). Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *arXiv preprint arXiv:2302.01018*.

Lotito, Q. F. and Montresor, A. (2020). Efficient algorithms to mine maximal span-trusses from temporal graphs. *arXiv preprint arXiv:2009.01928*.

Lü, L., Zhou, T., Zhang, Q.-M., and Stanley, H. E. (2016). The h-index of a network node and its relation to degree and coreness. *Nature communications*, 7(1):10168.

Ma, S., Hu, R., Wang, L., Lin, X., and Huai, J. (2020). An efficient approach to finding dense temporal subgraphs. *IEEE Transactions on Knowledge and Data Engineering*, 32(4):645–658.

Mackey, P., Porterfield, K., Fitzhenry, E., Choudhury, S., and Chin, G. (2018). A chronological edge-driven approach to temporal subgraph isomorphism. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE.

Malliaros, F. D., Giatsidis, C., Papadopoulos, A. N., and Vazirgiannis, M. (2020). The core decomposition of networks: Theory, algorithms and applications. *The VLDB Journal*, 29(1):61–92.

Mantzaris, A. V. and Higham, D. J. (2016). Asymmetry through time dependency. *The European Physical Journal B*, 89(3):71.

Mertzios, G. B., Molter, H., Niedermeier, R., Zamaraev, V., and Zschoche, P. (2019). Computing maximum matchings in temporal graphs. *arXiv preprint arXiv:1905.05304*.

Mertzios, G. B., Molter, H., and Zamaraev, V. (2018). Sliding window temporal graph coloring. *arXiv preprint arXiv:1811.04753*.

Mertzios, G. B., Nikoletseas, S., Raptopoulos, C., and Spirakis, P. G. (2024). On the existence of δ-temporal cliques in random simple temporal graphs. *arXiv preprint arXiv:2404.07147*.

Michail, O. (2016). An introduction to temporal graphs: An algorithmic perspective. *Internet Mathematics*, 12(4):239–280.

Michail, O. and Spirakis, P. G. (2016). Traveling salesman problems in temporal graphs. *Theoretical Computer Science*, 634:1–23.

Muthukrishnan, S. et al. (2005). Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236.

Mutzel, P. and Oettershagen, L. (2019). On the enumeration of bicriteria temporal paths. In *Theory and Applications of Models of Computation: 15th Annual Conference, TAMC 2019, Kitakyushu, Japan, April 13–16, 2019, Proceedings 15*, pages 518–535. Springer.

Oettershagen, L., Konstantinidis, A. L., and Italiano, G. F. (2023a). Temporal network core decomposition and community search. *arXiv preprint arXiv:2309.11843*.

Oettershagen, L., Kriege, N. M., and Mutzel, P. (2023b). A higher-order temporal h-index for evolving networks. In Singh, A. K., Sun, Y., Akoglu, L., Gunopulos, D., Yan, X., Kumar, R., Ozcan, F., and Ye, J., editors, *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 1770–1782. ACM.

Oettershagen, L. and Mutzel, P. (2022). Computing top-k temporal closeness in temporal networks. *Knowledge and Information Systems*, 64(2):507–535.

Oettershagen, L. and Mutzel, P. (2023). An index for temporal closeness computation in evolving graphs. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 280–288. SIAM.

Paranjape, A., Benson, A. R., and Leskovec, J. (2017). Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 601–610. ACM.

Pashanasangi, N. and Seshadhri, C. (2021). Faster and generalized temporal triangle counting, via degeneracy ordering. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21. ACM.

Pellegrina, L., Riondato, M., and Vandin, F. (2019). Hypothesis testing and statistically-sound pattern mining. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19. ACM.

Porter, A., Mirzasoleiman, B., and Leskovec, J. (2022). Analytical models for motifs in temporal networks. In *Companion Proceedings of the Web Conference 2022*, WWW '22. ACM.

Prakash, B. A., Vreeken, J., and Faloutsos, C. (2012). Spotting culprits in epidemics: How many and which ones? In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 11–20. IEEE.

Preti, G., Rozenshtein, P., Gionis, A., and Velegrakis, Y. (2021). *Discovering Dense Correlated Subgraphs in Dynamic Networks*, pages 395–407. Springer International Publishing.

references XI

Pu, J., Wang, Y., Li, Y., and Zhou, X. (2023). Sampling algorithms for butterfly counting on temporal bipartite graphs.

Qin, H., Li, R.-H., Wang, G., Huang, X., Yuan, Y., and Yu, J. X. (2020). Mining stable communities in temporal networks by density-based clustering. *IEEE Transactions on Big Data*.

Qin, H., Li, R.-H., Yuan, Y., Dai, Y., and Wang, G. (2023). Densest periodic subgraph mining on large temporal graphs. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11259–11273.

Qin, H., Li, R.-H., Yuan, Y., Wang, G., Qin, L., and Zhang, Z. (2022). Mining bursting core in large temporal graphs. *Proceedings of the VLDB Endowment*.

Renaud, L. and Naoki, M. (2016). *A Guide To Temporal Networks*, volume 4. World Scientific.

Ribeiro, P., Paredes, P., Silva, M. E. P., Aparicio, D., and Silva, F. (2021). A survey on subgraph counting: Concepts, algorithms, and applications to network motifs and graphlets. *ACM Computing Surveys*, 54(2):1–36.

Rocha, L. E. and Masuda, N. (2014). Random walk centrality for temporal networks. *New Journal of Physics*, 16(6):063023.

Rodriguez, M. G., Balduzzi, D., and Schölkopf, B. (2011). Uncovering the temporal dynamics of diffusion networks. *arXiv preprint arXiv:1105.0697*.

Rodriguez, M. G. and Schölkopf, B. (2012). Influence maximization in continuous time diffusion networks. *arXiv preprint arXiv:1205.1682*.

Rossetti, G. and Cazabet, R. (2018). Community discovery in dynamic networks: a survey. *ACM computing surveys (CSUR)*, 51(2):1–37.

Rozenshtein, P., Bonchi, F., Gionis, A., Sozio, M., and Tatti, N. (2019). Finding events in temporal networks: segmentation meets densest subgraph discovery. *Knowledge and Information Systems*, 62(4):1611–1639.

Rozenshtein, P. and Gionis, A. (2016). Temporal pagerank. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 674–689. Springer.

references XII

Rymar, M., Molter, H., Nichterlein, A., and Niedermeier, R. (2021). Towards classifying the polynomial-time solvability of temporal betweenness centrality. In *Graph-Theoretic Concepts in Computer Science: 47th International Workshop, WG 2021, Warsaw, Poland, June 23–25, 2021, Revised Selected Papers 47*, pages 219–231. Springer.

Santoro, D. and Sarpe, I. (2022). Onbra: Rigorous estimation of the temporal betweenness centrality in temporal networks. In *Proceedings of the ACM Web Conference 2022*, pages 1579–1588.

Santoro, N., Quattrociocchi, W., Flocchini, P., Casteigts, A., and Amblard, F. (2011). Time-varying graphs and social network analysis: Temporal indicators and metrics. *arXiv preprint arXiv:1102.0629*.

Sarpe, I. (2023). *Efficient and Rigorous Algorithms for the Analysis of Large Temporal Networks*. PhD thesis, University of Padova.

Sarpe, I. and Vandin, F. (2021a). oden: Simultaneous approximation of multiple motif counts in large temporal networks. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21. ACM.

Sarpe, I. and Vandin, F. (2021b). *PRESTO: Simple and Scalable Sampling Techniques for the Rigorous Approximation of Temporal Motif Counts*, pages 145–153. Society for Industrial and Applied Mathematics.

Scellato, S., Leontiadis, I., Mascolo, C., Basu, P., and Zafer, M. (2011). Evaluating temporal robustness of mobile networks. *IEEE Transactions on Mobile Computing*, 12(1):105–117.

Seidman, S. B. (1983). Network structure and minimum degree. *Social networks*, 5(3):269–287.

Semertzidis, K., Pitoura, E., Terzi, E., and Tsaparas, P. (2016). Best friends forever (bff): Finding lasting dense subgraphs. *arXiv preprint arXiv:1612.05440*.

Seshadhri, C. and Tirthapura, S. (2019). Scalable subgraph counting: The methods behind the madness. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19. ACM.

Shah, D. and Zaman, T. (2011). Rumors in a network: Who's the culprit? *IEEE Transactions on information theory*, 57(8):5163–5181.

references XIII

Shakarian, P., Bhatnagar, A., Aleali, A., Shaabani, E., and Guo, R. (2015). *Diffusion in social networks*. Springer.

Starnini, M., Machens, A., Cattuto, C., Barrat, A., and Pastor-Satorras, R. (2013). Immunization strategies for epidemic processes in time-varying contact networks. *Journal of theoretical biology*, 337:89–100.

Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Jin, D., Sheng, Q. Z., and Yu, P. S. (2024). A comprehensive survey on community detection with deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4682–4702.

Tang, J., Musolesi, M., Mascolo, C., and Latora, V. (2009). Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 31–36. ACM.

Tang, J., Musolesi, M., Mascolo, C., Latora, V., and Nicosia, V. (2010a). Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, pages 1–6.

Tang, J., Scellato, S., Musolesi, M., Mascolo, C., and Latora, V. (2010b). Small-world behavior in time-varying graphs. *Physical Review E*, 81(5):055101.

Tao, L., Kong, S., He, L., Zhang, F., Li, X., Jia, T., and Han, Z. (2022). A sequential-path tree-based centrality for identifying influential spreaders in temporal networks. *Chaos, Solitons & Fractals*, 165:112766.

Taylor, D., Porter, M. A., and Mucha, P. J. (2021). Tunable eigenvector-based centralities for multiplex and temporal networks. *Multiscale Modeling & Simulation*, 19(1):113–147.

Thejaswi, S., Lauri, J., and Gionis, A. (2020). Restless reachability problems in temporal graphs. *arXiv preprint arXiv:2010.08423*.

Thorup, M. (2000). Near-optimal fully-dynamic graph connectivity. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 343–350. Citeseer.

Tsalouchidou, I., Baeza-Yates, R., Bonchi, F., Liao, K., and Sellis, T. (2020). Temporal betweenness centrality in dynamic graphs. *International Journal of Data Science and Analytics*, 9:257–272.

Valdano, E., Poletto, C., Giovannini, A., Palma, D., Savini, L., and Colizza, V. (2015). Predicting epidemic risk from past temporal contact data. *PLoS computational biology*, 11(3):e1004152.

Viard, J., Latapy, M., and Magnien, C. (2015). Revealing contact patterns among high-school students using maximal cliques in link streams. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1517–1522.

Viard, T., Latapy, M., and Magnien, C. (2016). Computing maximal cliques in link streams. *Theoretical Computer Science*, 609:245–252.

Wang, C., Chen, W., and Wang, Y. (2012). Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576.

Wang, J., Wang, Y., Jiang, W., Li, Y., and Tan, K.-L. (2020). Efficient sampling algorithms for approximate temporal motif counting. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20. ACM.

Wen, D., Huang, Y., Zhang, Y., Qin, L., Zhang, W., and Lin, X. (2020). Efficiently answering span-reachability queries in large temporal graphs. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1153–1164. IEEE.

Wu, H., Cheng, J., Huang, S., Ke, Y., Lu, Y., and Xu, Y. (2014). Path problems in temporal graphs. *Proceedings of the VLDB Endowment*, 7(9):721–732.

Wu, H., Cheng, J., Lu, Y., Ke, Y., Huang, Y., Yan, D., and Wu, H. (2015). Core decomposition in large temporal graphs. In *Big Data*, pages 649–658. IEEE.

Xie, H., Fang, Y., Xia, Y., Luo, W., and Ma, C. (2023). On querying connected components in large temporal graphs. *Proceedings of the ACM on Management of Data*, 1(2):1–27.

Yang, J., Zhong, M., Zhu, Y., Qian, T., Liu, M., and Yu, J. X. (2023). Scalable time-range k-core query on temporal graphs. *arXiv preprint arXiv:2301.03770*.

references XV

Yu, M., Wen, D., Qin, L., Zhang, Y., Zhang, W., and Lin, X. (2021). On querying historical k-cores. *Proceedings of the VLDB Endowment*.

Yu, Y., Berger-Wolf, T. Y., Saia, J., et al. (2010). Finding spread blockers in dynamic networks. In *Advances in Social Network Mining and Analysis*, pages 55–76. Springer.

Yuan, Y., Ye, H., Vedula, S., Kaza, W., and Talati, N. (2023). Everest: Gpu-accelerated system for mining temporal motifs. *Proceedings of the VLDB Endowment*, 17(2):162–174.

Zaoli, S., Mazzarisi, P., and Lillo, F. (2019). Trip centrality: walking on a temporal multiplex with non-instantaneous link travel time. *Scientific reports*, 9(1):10570.

Zhang, Y., Lin, L., Yuan, P., and Jin, H. (2022). Significant engagement community search on temporal networks: Concepts and algorithms.

Zhuang, H., Sun, Y., Tang, J., Zhang, J., and Sun, X. (2013). Influence maximization in dynamic social networks. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1313–1318. IEEE.