Temporalizing Multi-Digraphs via Linear-Size Balanced Bi-Trees

Laurent Viennot

joint work with :
Alkida Balliu, Filippo Brunelli, Pierluigi Crescenzi, Dennis Olivetti
and :
Stéphane Bessy, Stéphan Thomassé

Algorithmic Aspects of Temporal Graphs 2024

Outline

Introduction : temporal graphs

Problem : temporalization

An approximate solution

Main tool : a balanced-separator Lemma for digraphs

Outline

Introduction : temporal graphs

Problem : temporalization

An approximate solution

Main tool : a balanced-separator Lemma for digraphs

Outline

Introduction : temporal graphs

Problem : temporalization

An approximate solution

Main tool : a balanced-separator Lemma for digraphs

Outline

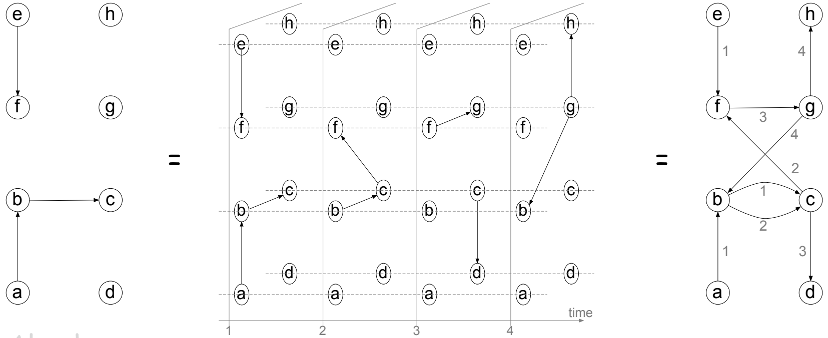
Introduction : temporal graphs

Problem : temporalization

An approximate solution

Main tool : a balanced-separator Lemma for digraphs

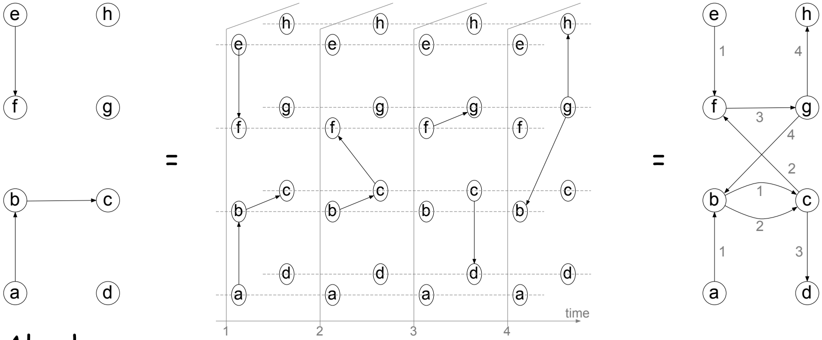
Temporal graphs (basic example)



Also known as :

- **time-dependent networks** [Cooke, Halsey 1966],
- **edge scheduled networks** [Berman 1996], **dynamic graphs** [Harary, Gupta 1997], **temporal networks** [Kempe, Kleinberg, Kumar 2002; Holme '15], **evolving graphs** [Bhadra, Ferreira '03],
- **time-varying graphs** [Casteigts, Flocchini, Quattrociocchi, Santoro 2012],
- **link streams** [Latapy, Viard, Magnien 2018],...

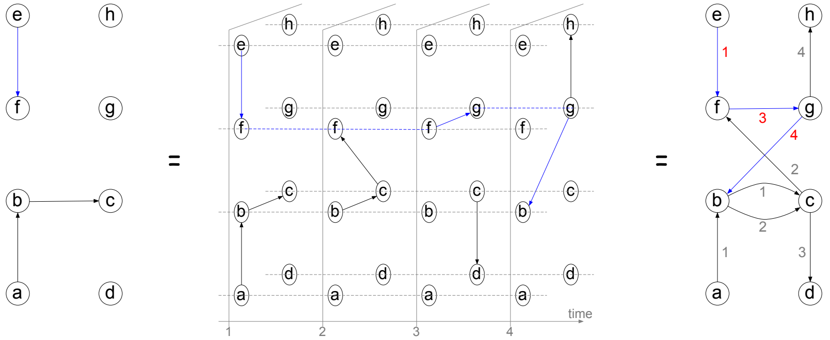
Temporal graphs (basic example)



Also known as :

- time-dependent networks [Cooke, Halsey 1966],
- edge scheduled networks [Berman 1996], dynamic graphs [Harary, Gupta 1997], temporal networks [Kempe, Kleinberg, Kumar 2002 ; Holme '15], evolving graphs [Bhadra, Ferreira '03],
- time-varying graphs [Casteigts, Flocchini, Quattrociocchi, Santoro 2012],
- link streams [Latapy, Viard, Magnien 2018],...

Temporal path/walk



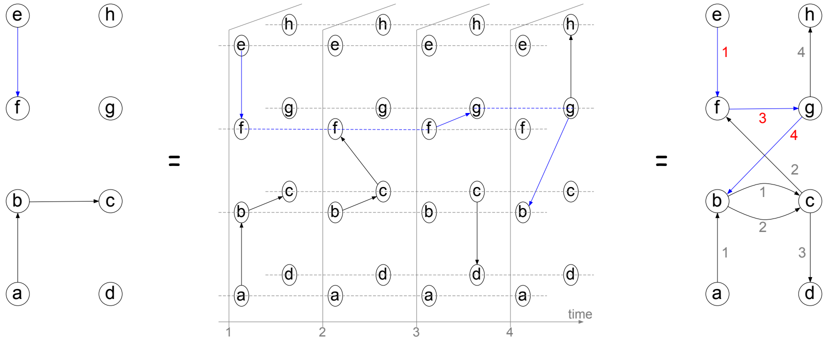
Definitions :

- Simple directed model : a temporal graph is a multi-digraph $D = (V, A)$ with a time labeling $\lambda : A \to \mathbb{N}$ of its edges.

- A path with edges $e_1, \dots, e_k$ is a (strict) temporal path when $\lambda(e_1) < \cdots < \lambda(e_k)$, i.e. time labels (strictly) increase along the path.

Example : $e \xrightarrow{1} f \xrightarrow{3} g \xrightarrow{4} b$ (waiting at f is allowed, $\neg a \longrightarrow f$)

Temporal path/walk

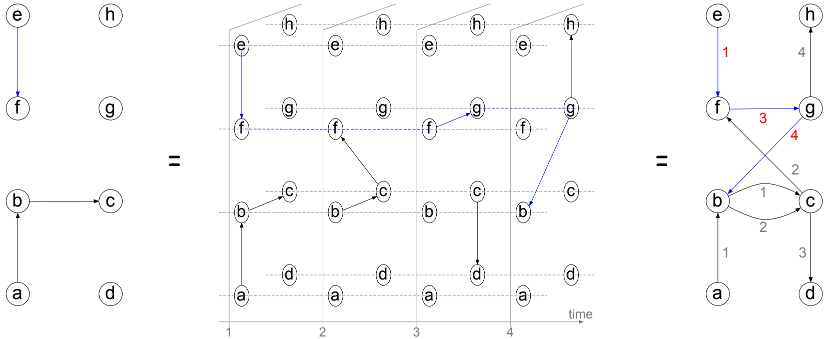


Definitions :

- Simple directed model : a temporal graph is a multi-digraph $D = (V, A)$ with a time labeling $\lambda : A \to \mathbb{N}$ of its edges.
- A path with edges $e_1, \dots, e_k$ is a (strict) temporal path when $\lambda(e_1) < \dots < \lambda(e_k)$, i.e. time labels (strictly) increase along the path.

Example : $e \xrightarrow{1} f \xrightarrow{3} g \xrightarrow{4} b$ (waiting at f is allowed, $\neg a \longrightarrow f$)

Temporal path/walk



Definitions :

- Simple directed model : a temporal graph is a multi-digraph $D = (V, A)$ with a time labeling $\lambda : A \to \mathbb{N}$ of its edges.
- A path with edges $e_1, \ldots, e_k$ is a (strict) temporal path when $\lambda(e_1) < \cdots < \lambda(e_k)$, i.e. time labels (strictly) increase along the path.

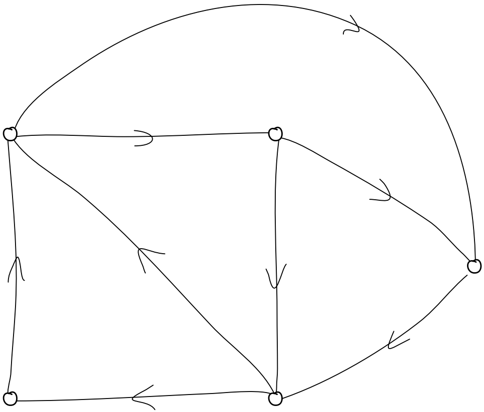
Example : $e \xrightarrow{1} f \xrightarrow{3} g \xrightarrow{4} b$ (waiting at f is allowed, $\neg a \longrightarrow f$)

Temporalization problem

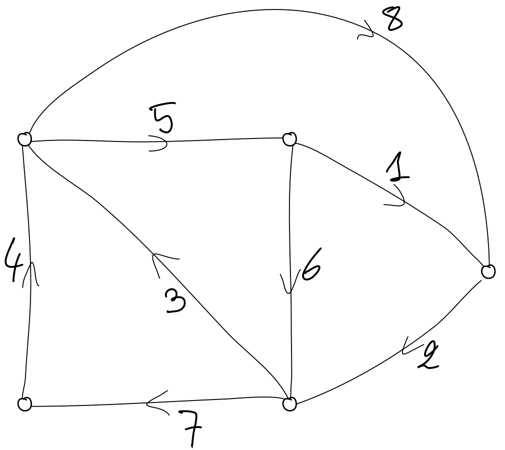
Directed graph temporalization

Problem : given a multi-digraph, assign time labels to edges so as to maximise the number of pairs temporally conneceted.

Example



Example



Motivation

Original motivation : optimize the schedule of a public transit network.

Also a natural fundamental problem.

The problem is mostly interesting when most pairs can be connected, we thus focus on strong digraphs.

We will see that this problem is related to fundamental properties of strong digraphs.

Motivation

Original motivation : optimize the schedule of a public transit network.

Also a natural fundamental problem.

The problem is mostly interesting when most pairs can be connected, we thus focus on strong digraphs.

We will see that this problem is related to fundamental properties of strong digraphs.

Motivation

Original motivation : optimize the schedule of a public transit network.

Also a natural fundamental problem.

The problem is mostly interesting when most pairs can be connected, we thus focus on strong digraphs.

We will see that this problem is related to fundamental properties of strong digraphs.

Motivation

Original motivation : optimize the schedule of a public transit network.

Also a natural fundamental problem.

The problem is mostly interesting when most pairs can be connected, we thus focus on strong digraphs.

We will see that this problem is related to fundamental properties of strong digraphs.

Related work

Undirected graph version (symmetric digraph, an arc and its reverse must have same time label) : deciding “label connectivity” is NP-complete [Göbel, Cerdeira, Veldman 1991].

Approximation is easy : use a spanning tree and a centroid node...

Minimizing $|\lambda|$, i.e. the number of labels, for achieving temporal connectivity is NP-hard [Klobas, Mertzios, Molter, Spirakis 2022].

It is NP-hard to decide if a strong digraph has a pair of edge-disjoint spanning in-tree and out-tree [Bang-Jensen 1991].

Related work

Undirected graph version (symmetric digraph, an arc and its reverse must have same time label) : deciding "label connectivity" is NP-complete [Göbel, Cerdeira, Veldman 1991].

Approximation is easy : use a spanning tree and a centroid node...

Minimizing $|\lambda|$, i.e. the number of labels, for achieving temporal connectivity is NP-hard [Klobas, Mertzios, Molter, Spirakis 2022].

It is NP-hard to decide if a strong digraph has a pair of edge-disjoint spanning in-tree and out-tree [Bang-Jensen 1991].

Related work

Undirected graph version (symmetric digraph, an arc and its reverse must have same time label) : deciding "label connectivity" is NP-complete [Göbel, Cerdeira, Veldman 1991].

Approximation is easy : use a spanning tree and a centroid node...

Minimizing $|\lambda|$, i.e. the number of labels, for achieving temporal connectivity is NP-hard [Klobas, Mertzios, Molter, Spirakis 2022].

It is NP-hard to decide if a strong digraph has a pair of edge-disjoint spanning in-tree and out-tree [Bang-Jensen 1991].

Related work

Undirected graph version (symmetric digraph, an arc and its reverse must have same time label) : deciding “label connectivity” is NP-complete [Göbel, Cerdeira, Veldman 1991].

Approximation is easy : use a spanning tree and a centroid node...

Minimizing $|\lambda|$, i.e. the number of labels, for achieving temporal connectivity is NP-hard [Klobas, Mertzios, Molter, Spirakis 2022].

It is NP-hard to decide if a strong digraph has a pair of edge-disjoint spanning in-tree and out-tree [Bang-Jensen 1991].

Hardness

Given a strong digraph D, deciding whether there exists an assignment of one time label per arc such that all pairs are temporally connected is NP-complete. [Balliu, Brunelli, Crescenzi, Olivetti, V. 2023]

Conjecture : any strong digraph has a pair of edge-disjtoint in-tree and out-tree both spanning $n/3$ nodes.

Hardness

Given a strong digraph D, deciding whether there exists an assignment of one time label per arc such that all pairs are temporally connected is NP-complete. [Balliu, Brunelli, Crescenzi, Olivetti, V. 2023]

Conjecture : any strong digraph has a pair of edge-disjtoint in-tree and out-tree both spanning $n/3$ nodes.

Problem variants

Problem 1 : Given a strong digraph, compute a temporalization connecting a constant fraction of pairs through strict temporal paths.

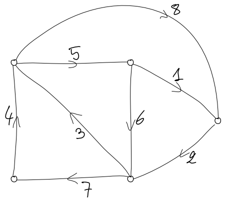
Problem 1bis edge-ordering (equivalent) : Compute an arc ordering σ for σ-respecting paths.

Problem node-ordering : a node ordering π for π-forward paths.

Problem variants

Problem 1 : Given a strong digraph, compute a temporalization connecting a constant fraction of pairs through strict temporal paths.

Problem 1bis edge-ordering (equivalent) : Compute an arc ordering σ for σ-respecting paths.

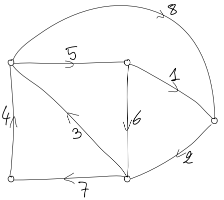


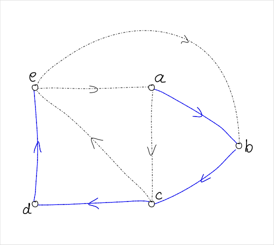
Problem node-ordering : a node ordering π for π-forward paths.

Problem variants

Problem 1 : Given a strong digraph, compute a temporalization connecting a constant fraction of pairs through strict temporal paths.

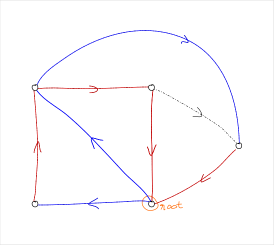
Problem 1bis edge-ordering (equivalent) : Compute an arc ordering σ for σ-respecting paths.





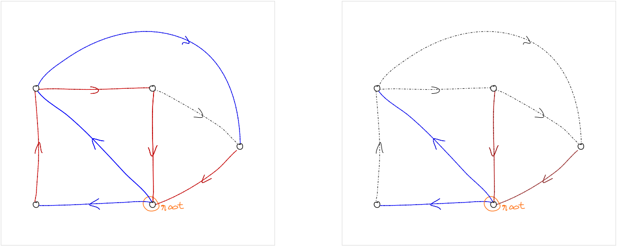
Problem node-ordering : a node ordering π for π-forward paths.

Problem 2 edge-disjoint trees (solves edge-ordering) :
Compute a pair of edge-disjoint in-tree and out-tree both spanning a constant fraction of nodes.



Problem 3 node-disjoint trees (solves all others) :
Compute a bitree, i.e. a pair of node-disjoint in-tree and out-tree both spanning a constant fraction of nodes.

Problem 2 edge-disjoint trees (solves edge-ordering) :
Compute a pair of edge-disjoint in-tree and out-tree both spanning a constant fraction of nodes.



Problem 3 node-disjoint trees (solves all others) :
Compute a bitree, i.e. a pair of node-disjoint in-tree and out-tree both spanning a constant fraction of nodes.

Main result : approximation

Any strong digraph D has a pair of node-disjtoint in-tree and out-tree both spanning $n/6$ nodes that can be computed in $O(n^2)$ time [Bessy, Thomassé, V. 2023].

Lemma : any strong digraph (V, A) has a balanced cyclic separator C, that is V can be partionned in I, C, O such that :

- C is spanned by a directed cycle,
- there are no arcs from I to O (directed separator),
- both $I \cup C$ and $I \cup O$ has size at least $n/3$ (balanced).

Main tool : a "left-maximal" DFS tree.

Main result : approximation

Any strong digraph D has a pair of node-disjtoint in-tree and out-tree both spanning $n/6$ nodes that can be computed in $O(n^2)$ time [Bessy, Thomassé, V. 2023].

Lemma : any strong digraph (V, A) has a balanced cyclic separator C, that is V can be partionned in I, C, O such that :

- C is spanned by a directed cycle,
- there are no arcs from I to O (directed separator),
- both $I \cup C$ and $I \cup O$ has size at least $n/3$ (balanced).

Main tool : a "left-maximal" DFS tree.

Main result : approximation

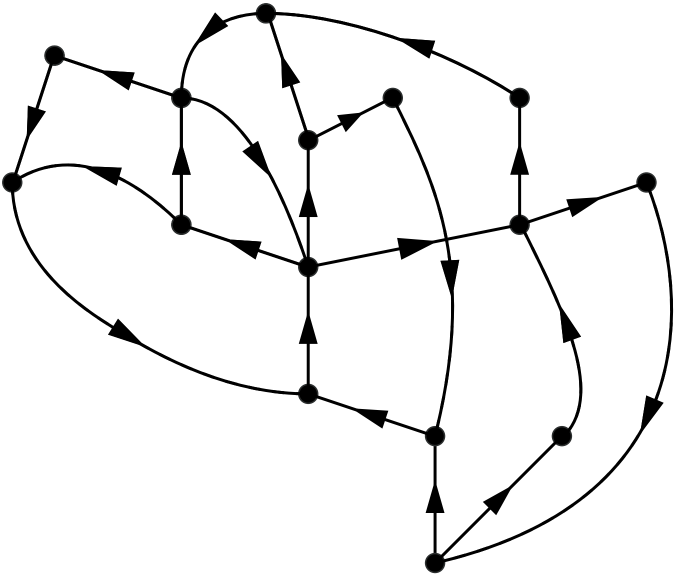
Any strong digraph D has a pair of node-disjtoint in-tree and out-tree both spanning $n/6$ nodes that can be computed in $O(n^2)$ time [Bessy, Thomassé, V. 2023].

Lemma : any strong digraph (V, A) has a balanced cyclic separator C, that is V can be partionned in I, C, O such that :

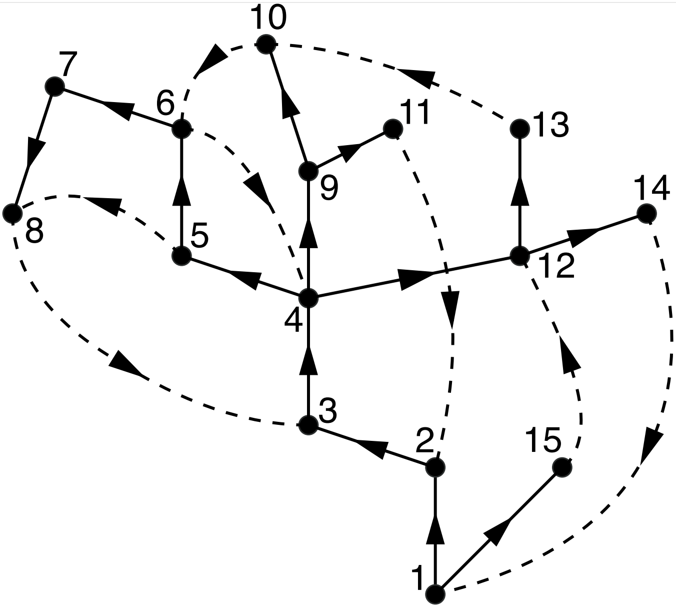
- C is spanned by a directed cycle,
- there are no arcs from I to O (directed separator),
- both $I \cup C$ and $I \cup O$ has size at least $n/3$ (balanced).

Main tool : a “left-maximal” DFS tree.

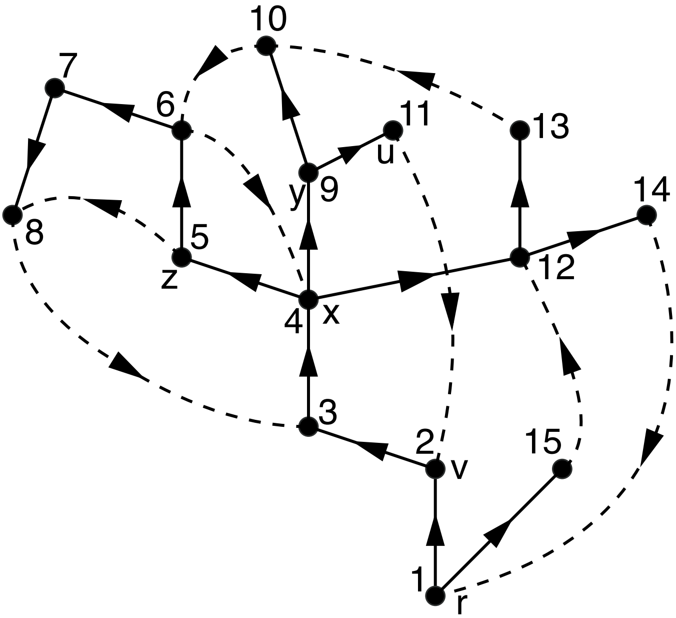
DFS to bitree



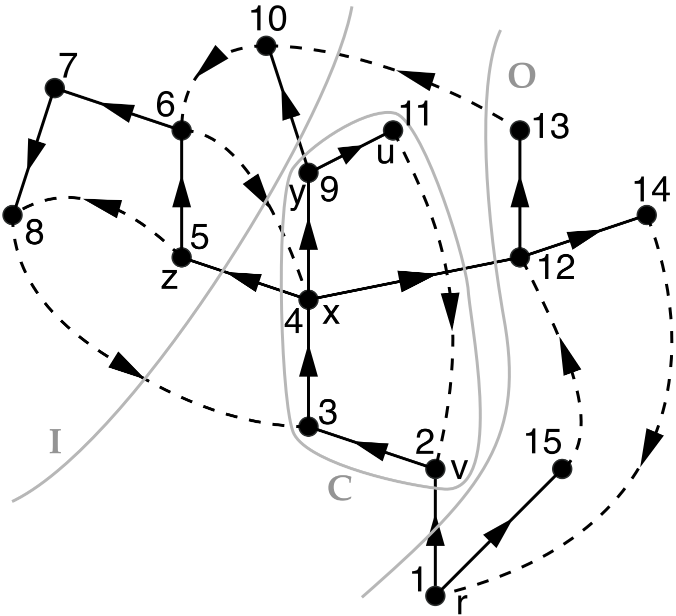
DFS to bitree



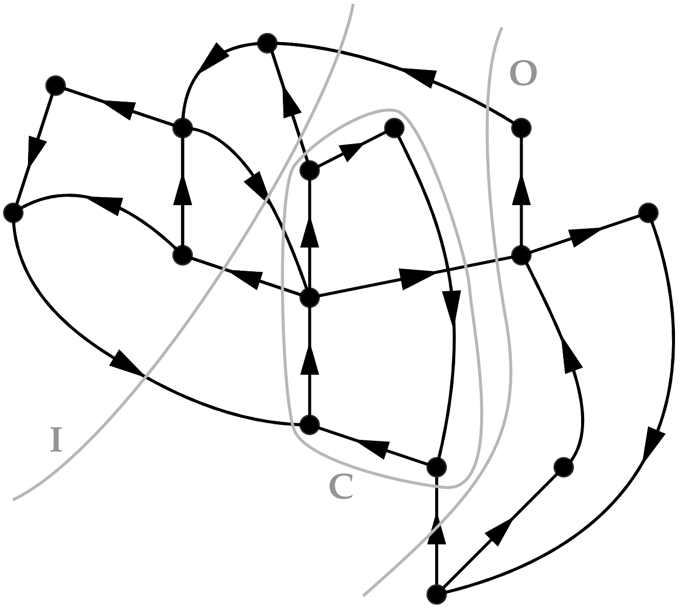
DFS to bitree



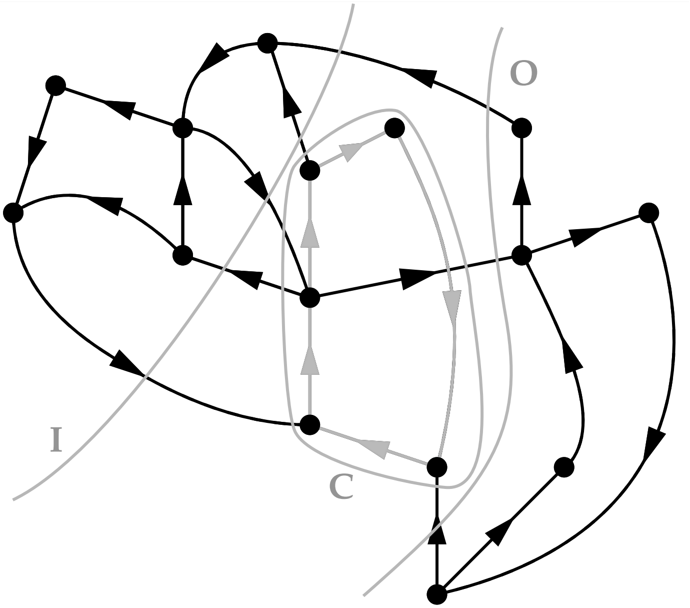
DFS to bitree



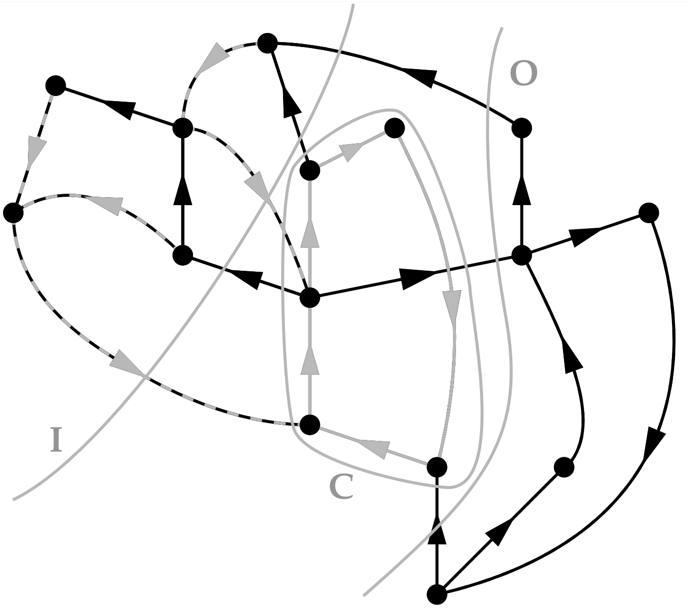
DFS to bitree



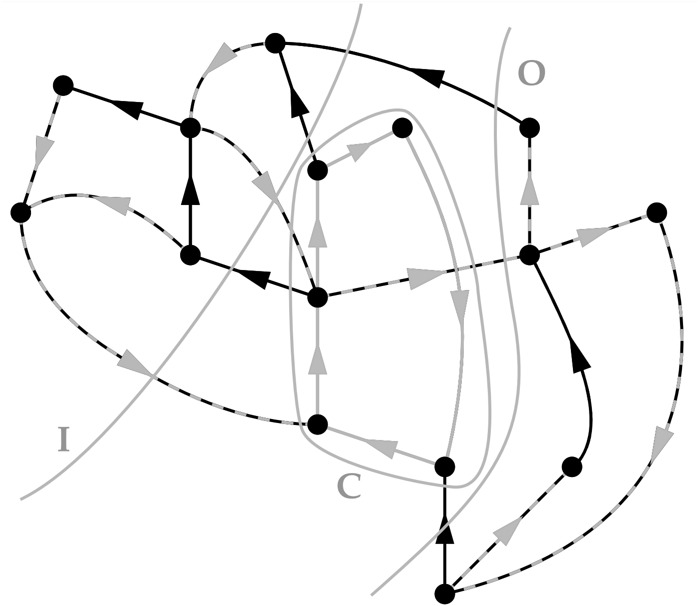
DFS to bitree



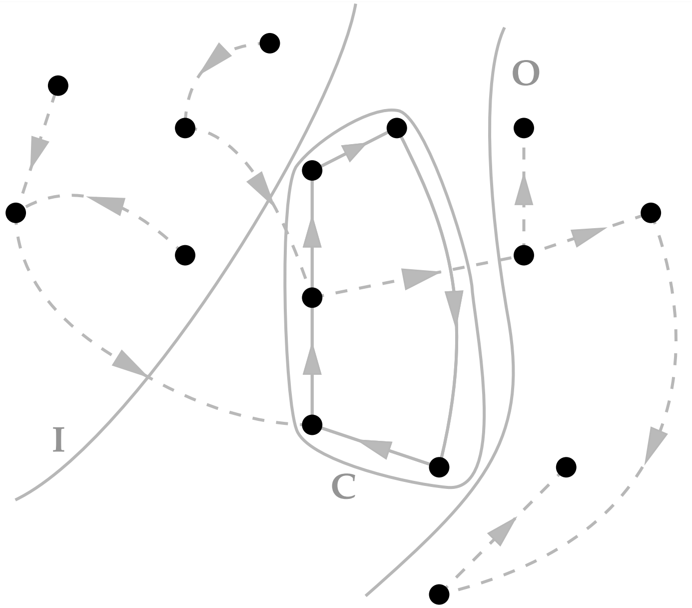
DFS to bitree



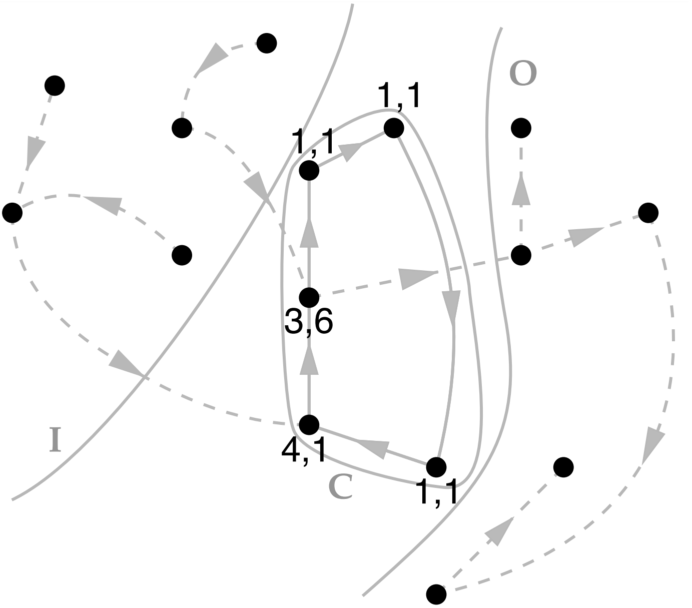
DFS to bitree



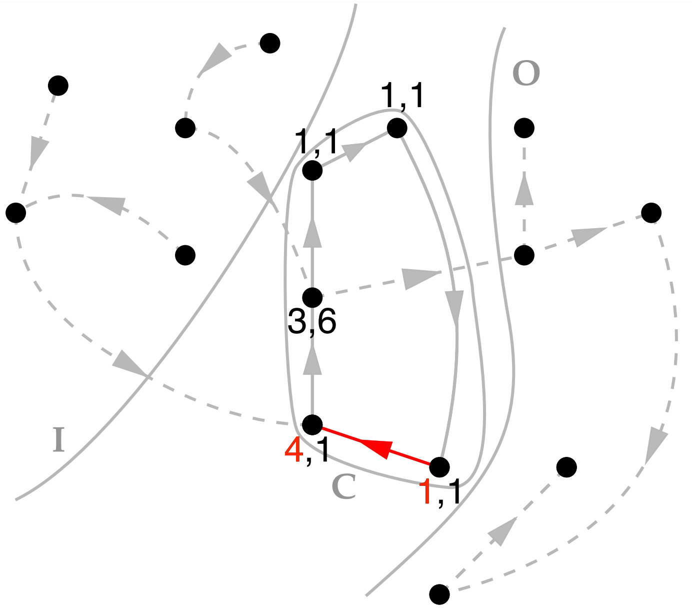
DFS to bitree



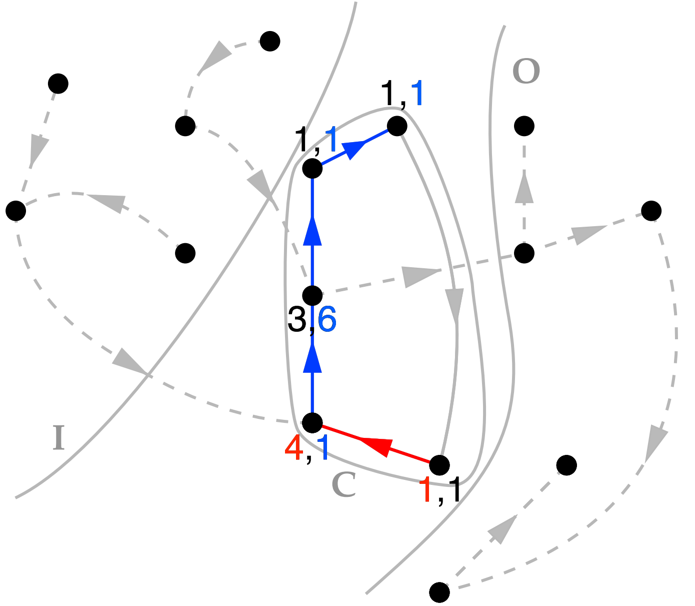
DFS to bitree



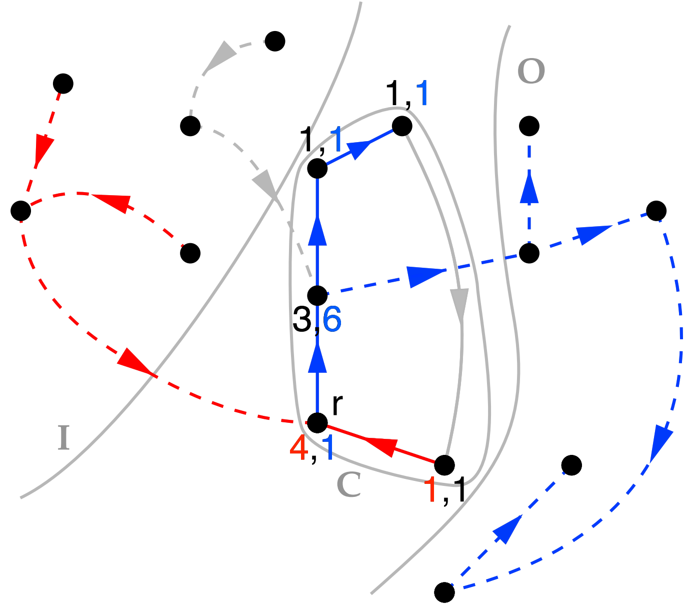
DFS to bitree



DFS to bitree



DFS to bitree



Extensions

The bitree construction generalizes to node weights.

And thus to a subset of nodes U (only pairs in $\binom{U}{2}$ are counted).

But not to an arbitrary set $R \subset \binom{V}{2}$ of requested pairs.

Conjecture : every strong digraph has a $O(\log n)$ forward cover, i.e. $k = O(\log n)$ node orderings such that any pair $\{x, y\}$ is connected by a path respecting one of the k orderings.

Transit networks : trip temporalization [Brunelli, Crescenzi, V. 2023].

Extensions

The bitree construction generalizes to node weights.

And thus to a subset of nodes U (only pairs in $\left(\binom{U}{2}\right)$ are counted).

But not to an arbitrary set $R \subset \left(\binom{V}{2}\right)$ of requested pairs.

Conjecture : every strong digraph has a $O(\log n)$ forward cover, i.e. $k = O(\log n)$ node orderings such that any pair $\{x, y\}$ is connected by a path respecting one of the k orderings.

Transit networks : trip temporalization [Brunelli, Crescenzi, V. 2023].

Extensions

The bitree construction generalizes to node weights.

And thus to a subset of nodes U (only pairs in $\left(\binom{U}{2}\right)$ are counted).

But not to an arbitrary set $R \subset \left(\binom{V}{2}\right)$ of requested pairs.

Conjecture : every strong digraph has a $O(\log n)$ forward cover, i.e. $k = O(\log n)$ node orderings such that any pair $\{x, y\}$ is connected by a path respecting one of the k orderings.

Transit networks : trip temporalization [Brunelli, Crescenzi, V. 2023].

Extensions

The bitree construction generalizes to node weights.

And thus to a subset of nodes U (only pairs in $\left(\binom{U}{2}\right)$ are counted).

But not to an arbitrary set $R \subset \left(\binom{V}{2}\right)$ of requested pairs.

Conjecture : every strong digraph has a $O(\log n)$ forward cover, i.e. $k = O(\log n)$ node orderings such that any pair $\{x, y\}$ is connected by a path respecting one of the k orderings.

Transit networks : trip temporalization [Brunelli, Crescenzi, V. 2023].

Extensions

The bitree construction generalizes to node weights.

And thus to a subset of nodes U (only pairs in $\left(\binom{U}{2}\right)$ are counted).

But not to an arbitrary set $R \subset \left(\binom{V}{2}\right)$ of requested pairs.

Conjecture : every strong digraph has a $O(\log n)$ forward cover, i.e. $k = O(\log n)$ node orderings such that any pair $\{x, y\}$ is connected by a path respecting one of the k orderings.

Transit networks : trip temporalization [Brunelli, Crescenzi, V. 2023].

Perspectives

What about general (non-strong) digraphs?

Is there a gap between edge ordering and node ordering? ($(n/3)^2$ vs $(n/6)^2$?)

What is the complexity of left-maximal DFS?

What about limited lifetime?

Perspectives

What about general (non-strong) digraphs?

Is there a gap between edge ordering and node ordering?
($(n/3)^2$ vs $(n/6)^2$?)

What is the complexity of left-maximal DFS?

What about limited lifetime?

Perspectives

What about *general* (non-strong) digraphs?

Is there a gap between edge ordering and node ordering?
($(n/3)^2$ vs $(n/6)^2$?)

What is the complexity of left-maximal DFS?

What about limited lifetime?

Perspectives

What about general (non-strong) digraphs?

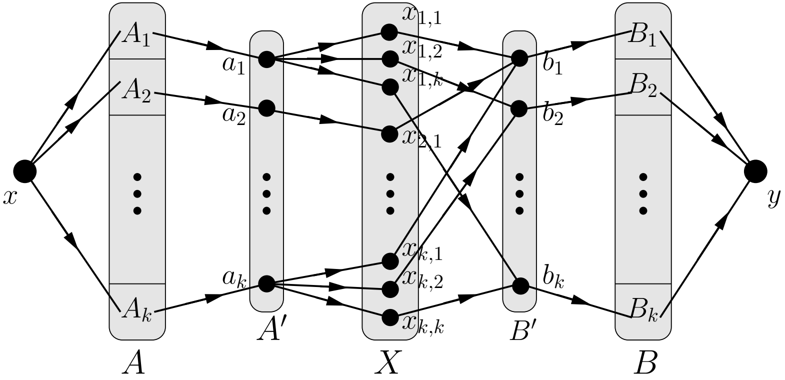
Is there a gap between edge ordering and node ordering?
($(n/3)^2$ vs $(n/6)^2$?)

What is the complexity of left-maximal DFS?

What about limited lifetime?

Thanks.

A good node ordering may give poor bitrees.



Open problem

MRET is in APX for strong digraphs.

Does it hold also for general digraphs?

Open problem

MRET is in APX for strong digraphs.

Does it hold also for general digraphs?

Edge-disjoint in-tree and out-tree : related work

Related problem : Find an in-tree and an out-tree with same root that are edge disjoint and have size $\Omega(n)$.

It is NP-hard to decide if a strong digraph has such a pair. [Bang-Jensen 1991]

Conjecture : There exist c such that any c-edge-connected digraph has such a pair. [Thomassen 1989]

Th : Every strong digraph has an in-tree and an out-tree with same root that are vertex disjoint and both have size $n/6$. [Bessy, Thomassé, Viennot 2023]

Question : How many pairs of in-tree, out-tree are needed to cover all pairs ?

Edge-disjoint in-tree and out-tree : related work

Related problem : Find an in-tree and an out-tree with same root that are edge disjoint and have size $\Omega(n)$.

It is NP-hard to decide if a strong digraph has such a pair.
[Bang-Jensen 1991]

Conjecture : There exist c such that any c-edge-connected digraph has such a pair. [Thomassen 1989]

Th : Every strong digraph has an in-tree and an out-tree with same root that are vertex disjoint and both have size $n/6$. [Bessy, Thomassé, Viennot 2023]

Question : How many pairs of in-tree, out-tree are needed to cover all pairs ?

Edge-disjoint in-tree and out-tree : related work

Related problem : Find an in-tree and an out-tree with same root that are edge disjoint and have size $\Omega(n)$.

It is NP-hard to decide if a strong digraph has such a pair. [Bang-Jensen 1991]

Conjecture : There exist c such that any c-edge-connected digraph has such a pair. [Thomassen 1989]

Th : Every strong digraph has an in-tree and an out-tree with same root that are vertex disjoint and both have size $n/6$. [Bessy, Thomassé, Viennot 2023]

Question : How many pairs of in-tree, out-tree are needed to cover all pairs ?

Edge-disjoint in-tree and out-tree : related work

Related problem : Find an in-tree and an out-tree with same root that are edge disjoint and have size $\Omega(n)$.

It is NP-hard to decide if a strong digraph has such a pair. [Bang-Jensen 1991]

Conjecture : There exist c such that any c-edge-connected digraph has such a pair. [Thomassen 1989]

Th : Every strong digraph has an in-tree and an out-tree with same root that are vertex disjoint and both have size $n/6$. [Bessy, Thomassé, Viennot 2023]

Question : How many pairs of in-tree, out-tree are needed to cover all pairs ?

Edge-disjoint in-tree and out-tree : related work

Related problem : Find an in-tree and an out-tree with same root that are edge disjoint and have size $\Omega(n)$.

It is NP-hard to decide if a strong digraph has such a pair. [Bang-Jensen 1991]

Conjecture : There exist c such that any c-edge-connected digraph has such a pair. [Thomassen 1989]

Th : Every strong digraph has an in-tree and an out-tree with same root that are vertex disjoint and both have size $n/6$. [Bessy, Thomassé, Viennot 2023]

Question : How many pairs of in-tree, out-tree are needed to cover all pairs ?

Undirected graph temporalization is quite known

Given an undirected graph G, deciding whether there exists an assignment of one time label per edge such that all pairs are temporally connected is NP-complete. [Göbel, Cerdeira, Veldman 1991]

Approximation is obvious : take a spanning tree and assign time labels that connect $(n/2)^2$ pairs.

Related (Gossip/telephone problem [Bumby 1981]) : The minimum number of time labels allowing to temporally connect all pairs at least $2n-4$, and equals $2n-4$ if G has a C_4 (one or two time labels per edge).

Undirected graph temporalization is quite known

Given an undirected graph G, deciding whether there exists an assignment of one time label per edge such that all pairs are temporally connected is NP-complete. [Göbel, Cerdeira, Veldman 1991]

Approximation is obvious : take a spanning tree and assign time labels that connect $(n/2)^2$ pairs.

Related (Gossip/telephone problem [Bumby 1981]) : The minimum number of time labels allowing to temporally connect all pairs at least $2n-4$, and equals $2n-4$ if G has a C_4 (one or two time labels per edge).

Undirected graph temporalization is quite known

Given an undirected graph G, deciding whether there exists an assignment of one time label per edge such that all pairs are temporally connected is NP-complete. [Göbel, Cerdeira, Veldman 1991]

Approximation is obvious : take a spanning tree and assign time labels that connect $(n/2)^2$ pairs.

Related (Gossip/telephone problem [Bumby 1981]) : The minimum number of time labels allowing to temporally connect all pairs at least $2n-4$, and equals $2n-4$ if G has a C_4 (one or two time labels per edge).

Temporal graph models

Main models (interval availability)

Time-dependent network : a graph where edge delay depends on time [Cooke, Halsey 1966] :
$\mathcal{G} = ((V, E), \delta)$ with $\delta : E \to \mathbb{R}^{\mathbb{R}}$ ($\delta(e)(\tau)$ is the delay of $e \in E$ at time τ).

Time-varying graph : edge and nodes are available at certain times.

Pice-wise constant-delay : each $\delta(e)$ is piecewise constant.
[Bui-Xuan, Ferreira, Jarry 2003], [Dehne, Omran, Sack 2012]

Pice-wise zero-delay : $\delta(e)$ has value 0 or ∞.
Temporal networks [Holme 2015], link-stream [Latapy, Viard, Magnien 2018].

Main models (interval availability)

Time-dependent network : a graph where edge delay depends on time [Cooke, Halsey 1966] :
$\mathcal{G} = ((V, E), \delta)$ with $\delta : E \to \mathbb{R}^{\mathbb{R}}$ ($\delta(e)(\tau)$ is the delay of $e \in E$ at time τ).

Time-varying graph : edge and nodes are available at certain times.

Pice-wise constant-delay : each $\delta(e)$ is piecewise constant.
[Bui-Xuan, Ferreira, Jarry 2003], [Dehne, Omran, Sack 2012]

Pice-wise zero-delay : $\delta(e)$ has value 0 or ∞.
Temporal networks [Holme 2015], link-stream [Latapy, Viard, Magnien 2018].

Main models (interval availability)

Time-dependent network : a graph where edge delay depends on time [Cooke, Halsey 1966] : $\mathcal{G} = ((\mathsf{V}, \mathsf{E}), \delta)$ with $\delta : \mathsf{E} \to \mathbb{R}^{\mathbb{R}}$ ($\delta(\mathsf{e})(\tau)$ is the delay of $\mathsf{e} \in \mathsf{E}$ at time τ).

Time-varying graph : edge and nodes are available at certain times.

Pice-wise constant-delay : each $\delta(\mathsf{e})$ is piecewise constant. [Bui-Xuan, Ferreira, Jarry 2003], [Dehne, Omran, Sack 2012]

Pice-wise zero-delay : $\delta(\mathsf{e})$ has value 0 or ∞.
Temporal networks [Holme 2015], link-stream [Latapy, Viard, Magnien 2018].

Main models (interval availability)

Time-dependent network : a graph where edge delay depends on time [Cooke, Halsey 1966] :
$\mathcal{G} = ((\mathsf{V}, \mathsf{E}), \delta)$ with $\delta : \mathsf{E} \to \mathbb{R}^{\mathbb{R}}$ ($\delta(\mathsf{e})(\tau)$ is the delay of $\mathsf{e} \in \mathsf{E}$ at time τ).

Time-varying graph : edge and nodes are available at certain times.

Pice-wise constant-delay : each $\delta(\mathsf{e})$ is piecewise constant. [Bui-Xuan, Ferreira, Jarry 2003], [Dehne, Omran, Sack 2012]

Pice-wise zero-delay : $\delta(\mathsf{e})$ has value 0 or ∞.
Temporal networks [Holme 2015], link-stream [Latapy, Viard, Magnien 2018].

Main models (point availability)

Evolving graph : sequence of (static) graphs with same vertices $\mathcal{G} = (\mathsf{V}, \mathsf{E}_1, \ldots, \mathsf{E}_\ell)$ [Bhadra, Ferreira 2003] or equivalently time labeled graphs $\mathcal{G} = ((\mathsf{E}, \mathsf{V}), \lambda)$ with $\lambda : \mathsf{E} \to 2^{\mathbb{N}}$ [Michail 2016].
(Delay of edges is 1/0 for strict/non-strict temporal paths.)

Simple : $\mathcal{G} = ((\mathsf{E}, \mathsf{V}), \lambda)$ with $\lambda : \mathsf{E} \to \mathbb{N}$ [Kempe, Kleinberg, Kumar 2002].
Proper : edges incident to a node have pairwise disjoint labels (strict and non-strict then coincide) [Casteigts, Corsini, Sarkar 2022].
Happy : simple and proper [Casteigts, Corsini, Sarkar 2022].
Globally happy : simple and pairwise-disjoint time labels.

Main models (point availability)

Evolving graph : sequence of (static) graphs with same vertices $\mathcal{G} = (\mathsf{V}, \mathsf{E}_1, \dots, \mathsf{E}_\ell)$ [Bhadra, Ferreira 2003]
or equivalently time labeled graphs $\mathcal{G} = ((\mathsf{E}, \mathsf{V}), \lambda)$ with $\lambda : \mathsf{E} \to 2^{\mathbb{N}}$ [Michail 2016].
(Delay of edges is 1/0 for strict/non-strict temporal paths.)

Simple : $\mathcal{G} = ((\mathsf{E}, \mathsf{V}), \lambda)$ with $\lambda : \mathsf{E} \to \mathbb{N}$ [Kempe, Kleinberg, Kumar 2002].
Proper : edges incident to a node have pairwise disjoint labels (strict and non-strict then coincide) [Casteigts, Corsini, Sarkar 2022].
Happy : simple and proper [Casteigts, Corsini, Sarkar 2022].
Globally happy : simple and pairwise-disjoint time labels.

What is time?

Time domain : $\mathbb{N}$ or $\mathbb{R}$ or $\mathbb{T}$?

Discrete :

- time is discrete,
- and/or edges are available at some given points in time,
- and/or traversal takes time 1 (or 0).

Continous :

- time is continuous,
- and/or edges are available during given intervals of time,
- and/or traversal takes constant time.

What is time?

Time domain : $\mathbb{N}$ or $\mathbb{R}$ or $\mathbb{T}$?

Discrete :

- time is discrete,
- and/or edges are available at some given points in time,
- and/or traversal takes time 1 (or 0).

Continous :

- time is continuous,
- and/or edges are available during given intervals of time,
- and/or traversal takes constant time.

What is time?

Time domain : $\mathbb{N}$ or $\mathbb{R}$ or $\mathbb{T}$?

Discrete :

- time is discrete,
- and/or edges are available at some given points in time,
- and/or traversal takes time 1 (or 0).

Continous :

- time is continuous,
- and/or edges are available during given intervals of time,
- and/or traversal takes constant time.

What is time?

Time domain : $\mathbb{N}$ or $\mathbb{R}$ or $\mathbb{T}$?

Discrete :

- time is discrete,
- and/or edges are available at some given points in time,
- and/or traversal takes time 1 (or 0).

Continous :

- time is continuous,
- and/or edges are available during given intervals of time,
- and/or traversal takes constant time.