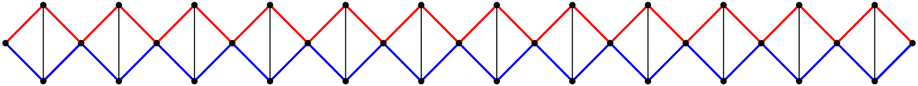
Path Covers of Temporal Graphs: When is Dilworth dynamic?

Dibyayan Chakraborty
School of Computing, University of Leeds

Algorithmic Aspects of Temporal Graphs VII
Satellite workshop of ICALP 2024
07-07-2024



Joint work with

Figure: (from left) Dr. Antoine Dailly, Dr. Florent Foucaud, Prof. Ralf Klasing

- LIMOS, University of Clermont-Ferrand, France.
- School of Computing, University of Leeds, UK.
- LaBRI, University of Bordeaux.

Thanks to Antoine for the slides !

- Path Cover: A set of Paths that Cover all Vertices of a (di)graph.

Path Cover Problem

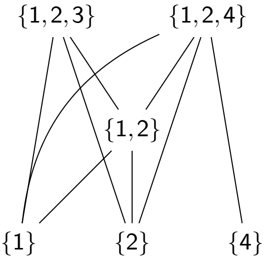
Input: A (di)graph

Output: A Path Cover of min. Size.

→ Hard in general.

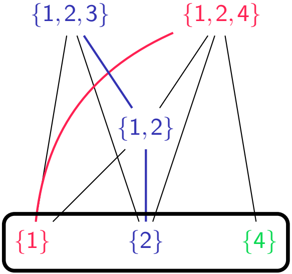
→ Polynomial time Solvable in DAGs.

Introduction: Dilworth's theorem



Theorem [Dilworth, 1950]

The minimum size of a chain partition of a finite poset is equal to the maximum size of an antichain of this poset.

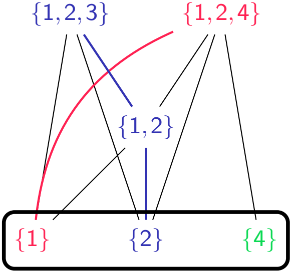


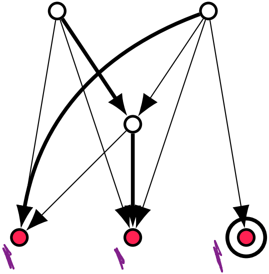
Introduction: Dilworth's theorem

Theorem [Dilworth, 1950]

The minimum size of a path partition of a transitive DAG is equal to the maximum size of an antichain of this DAG.

Restated for graphs...



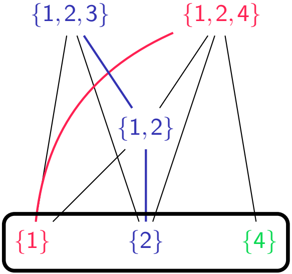


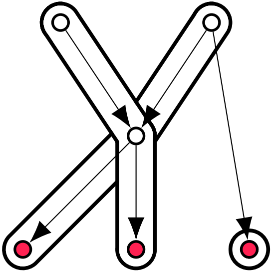
Introduction: Dilworth's theorem

Theorem [Dilworth, 1950]

The minimum size of a path cover of a DAG is equal to the maximum size of an antichain of this DAG.

Restated for graphs...
... and covers.





Theorem [Dilworth, 1950]

The minimum size of a path cover of a DAG is equal to the maximum size of an antichain of this DAG.

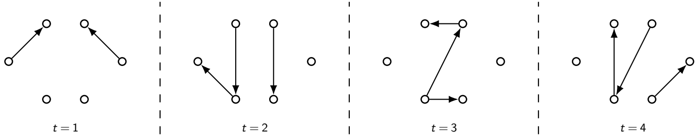
Restated for graphs...
... and covers.

Algorithms:

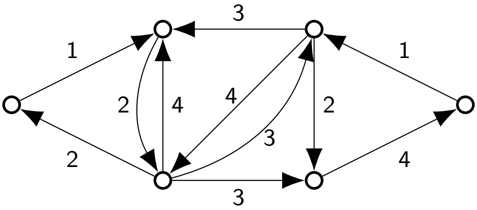
- Algorithmic proof (polynomial time) [Fulkerson, 1956]

A temporal analogue of Dilworth's theorem?

$$\mathscr{D} = (V, A_1, A_2, \ldots, A_k)$$

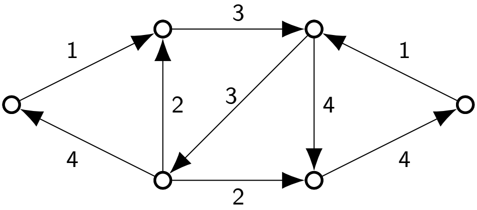


$$\mathscr{D} = (V, A, \lambda)$$



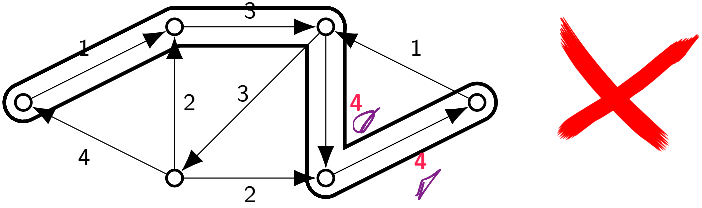
Many results and applications in distributed algorithms, dynamic networks (transportation, social, biological...), interest in the graph algorithms community.

- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).

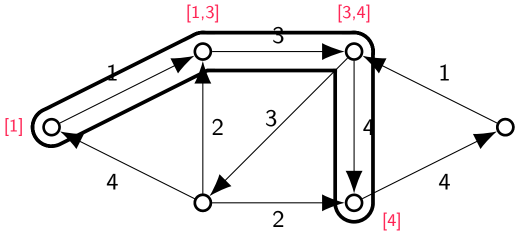


A few definitions for this talk

- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.

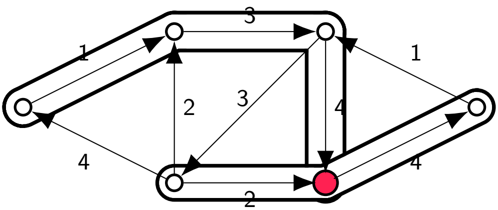


- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.
- A temporal path occupies a vertex during interval $[t_1, t_2]$ if it reaches it at time t_1 and leaves it at time t_2.

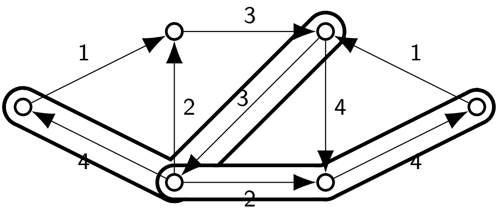


A few definitions for this talk

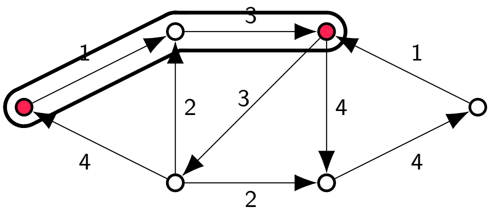
- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.
- A temporal path occupies a vertex during interval $[t_1, t_2]$ if it reaches it at time t_1 and leaves it at time t_2.
- Two temporal paths intersect if they occupy the same vertex during non-disjoint intervals.



- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.
- A temporal path occupies a vertex during interval $[t_1, t_2]$ if it reaches it at time t_1 and leaves it at time t_2.
- Two temporal paths intersect if they occupy the same vertex during non-disjoint intervals. They are temporally disjoint if they do not intersect.

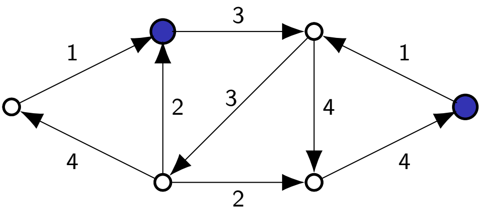


- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.
- A temporal path occupies a vertex during interval $[t_1, t_2]$ if it reaches it at time t_1 and leaves it at time t_2.
- Two temporal paths intersect if they occupy the same vertex during non-disjoint intervals. They are temporally disjoint if they do not intersect.
- Two vertices are temporally connected if there is a temporal path between them.



A few definitions for this talk

- A temporal DAG (resp. tree...) is a temporal (di)graph whose underlying (di)graph is a DAG (resp. tree...).
- (Directed) temporal path : strictly increasing time labels.
- A temporal path occupies a vertex during interval $[t_1, t_2]$ if it reaches it at time t_1 and leaves it at time t_2.
- Two temporal paths intersect if they occupy the same vertex during non-disjoint intervals. They are temporally disjoint if they do not intersect.
- Two vertices are temporally connected if there is a temporal path between them.
- A temporal antichain is a set of vertices who are pairwise not temporally connected.

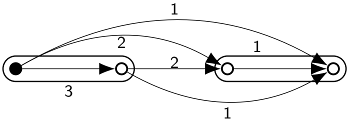


A temporal Dilworth's theorem?

Temporal Dilworth property?

In a temporal DAG, the minimum size of a temporal path partition/cover is equal to the maximum size of a temporal antichain.

Not all temporal DAGs have the Temporal Dilworth Property.



A temporal Dilworth's theorem?

Temporal Dilworth property?

In a temporal DAG, the minimum size of a temporal path partition/cover is equal to the maximum size of a temporal antichain.

Two problems:

Temporal Path Cover (TPC)

Temporal Path Partition/Temporally Disjoint Path Cover (TD-PC)

Two questions:

Which temporal DAGs have the Dilworth property?

$\Rightarrow$ **Combinatorial** aspect

What is the complexity of those problems?

$\Rightarrow$ **Algorithmic** aspect

Our results

⟨Temporal Path Cover⟩ ⟨Temporally - Disj. Path Cover⟩

Temporal class	**TPC**	**TD-PC**
Oriented paths	$\mathcal{O}(\ell n)$	$\mathcal{O}(\ell n)$
Rooted trees	$\mathcal{O}(\ell n^2)$	$\mathcal{O}(\ell n^2)$
Oriented trees	$\mathcal{O}(\ell n^2 + n^3)$	NP-hard
DAGs*	NP-hard	NP-hard
Digraphs	XP (tw and $t_{\max}$) $n^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}$	FPT (tw and $t_{\max}$) $2^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}\, n$

* planar, subcubic, bipartite, girth 10, $\ell = 1$, $t_{\max} = 2$

$$n = \text{number of vertices}$$
$$\ell = \text{number of (unsorted) time labels per arc}$$
$$t_{\max} = \text{total number of time-steps}$$

Classes with polynomial-time algorithm also have the Dilworth property.

⟨Temporal Path Cover⟩ ⟨Temporally-Disj. Path Cover⟩

Temporal class	TPC	TD-PC
Oriented paths	$\mathcal{O}(\ell n)$	$\mathcal{O}(\ell n)$
Rooted trees	$\mathcal{O}(\ell n^2)$	$\mathcal{O}(\ell n^2)$
Oriented trees	$\mathcal{O}(\ell n^2 + n^3)$	NP-hard
DAGs*	NP-hard	NP-hard
Digraphs	XP (tw and $t_{\max}$) $n^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}$	FPT (tw and $t_{\max}$) $2^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}\, n$

Not difficult

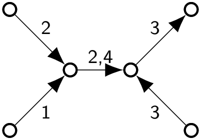
* planar, subcubic, bipartite, girth 10, $\ell = 1$, $t_{\max} = 2$

$$n = \text{number of vertices}$$
$$\ell = \text{number of (unsorted) time labels per arc}$$
$$t_{\max} = \text{total number of time-steps}$$

Classes with polynomial-time algorithm also have the Dilworth property.

Theorem [CDFK, 2024+]

Temporal oriented trees have the Dilworth property for TPC, and we can solve TPC in time $\mathcal{O}(\ell n^2 + n^3)$.

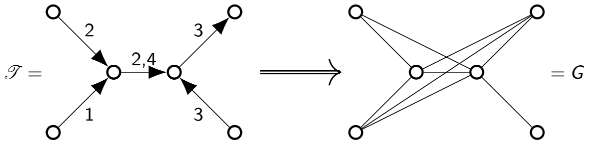
$\mathscr{T} =$ 

Theorem [CDFK, 2024+]

Temporal oriented trees have the Dilworth property for TPC, and we can solve TPC in time $\mathcal{O}(\ell n^2 + n^3)$.

Algorithm

- Construct an auxiliary connectivity graph: two vertices are adjacent $\Leftrightarrow$ they are temporally connected in the tree



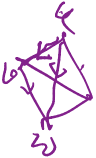
Min
- clique Cover in the Connectivity graph
- (Min clique Cover) $\Leftrightarrow$ (Min Path Cover) of the tree.

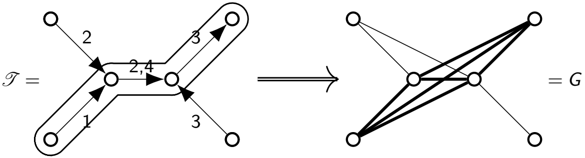
Theorem [CDFK, 2024+]

Temporal oriented trees have the Dilworth property for TPC, and we can solve TPC in time $\mathcal{O}(\ell n^2 + n^3)$.

Algorithm

- Construct an auxiliary connectivity graph: two vertices are adjacent $\Leftrightarrow$ they are temporally connected in the tree





Lemma

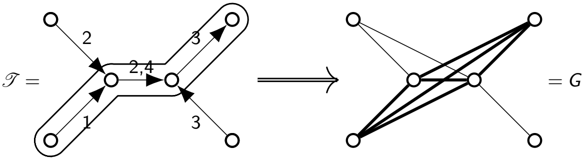
Clique in G $\Leftrightarrow$ Temporal Path in $\mathcal{T}$.

Theorem [CDFK, 2024+]

Temporal oriented trees have the Dilworth property for TPC, and we can solve TPC in time $\mathcal{O}(\ell n^2 + n^3)$.

Algorithm

- Construct an auxiliary connectivity graph: two vertices are adjacent $\Leftrightarrow$ they are temporally connected in the tree



Lemma

Clique Cover in G $\Leftrightarrow$ Temporal Path Cover in $\mathscr{T}$.

Path Cover of temporal oriented trees (2) Holes

Lemma

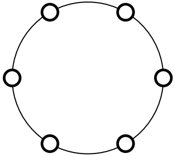
There are no holes of length greater than four in the connectivity graph.

Lemma

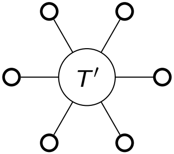
There are no holes of length greater than four in the connectivity graph.

Claim

The vertices of the hole are leaves of a connected subtree T'.



G



$\mathscr{T}$

Lemma

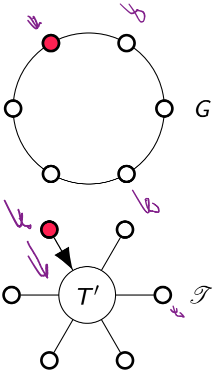
There are no holes of length greater than four in the connectivity graph.

Claim

The vertices of the hole are leaves of a connected subtree T'.

Claim

Alternating between in-arcs and out-arcs from and to T'.



Lemma

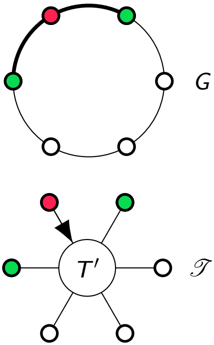
There are no holes of length greater than four in the connectivity graph.

Claim

The vertices of the hole are leaves of a connected subtree T'.

Claim

Alternating between in-arcs and out-arcs from and to T'.



Lemma

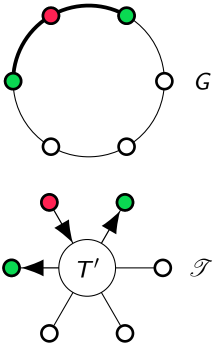
There are no holes of length greater than four in the connectivity graph.

Claim

The vertices of the hole are leaves of a connected subtree T'.

Claim

Alternating between in-arcs and out-arcs from and to T'.



Lemma

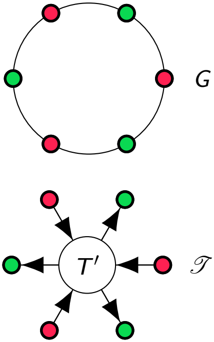
There are no holes of length greater than four in the connectivity graph.

Claim

The vertices of the hole are leaves of a connected subtree T'.

Claim

Alternating between in-arcs and out-arcs from and to T'.



Lemma

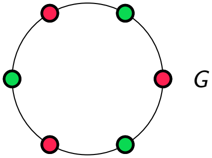
There are no holes of length greater than four in the connectivity graph.

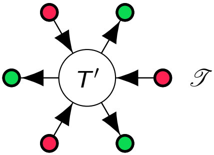
Claim

The vertices of the hole are leaves of a connected subtree T'.

Claim

Alternating between in-arcs and out-arcs from and to T'.
$\Rightarrow$ No odd hole





Lemma

There are no holes of length greater than four in the connectivity graph.

Claim

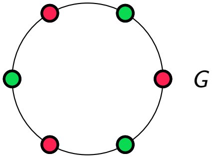
The vertices of the hole are leaves of a connected subtree T'.

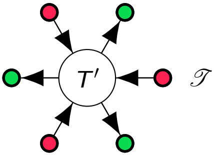
Claim

Alternating between in-arcs and out-arcs from and to T'.
$\Rightarrow$ No odd hole

Claim

No even holes of length greater than four either (using Helly property and vertex-intersection of temporal paths).





Lemma

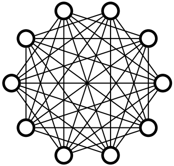
There are no antiholes in the connectivity graph.

Complement of holes ≥ 5

Lemma

There are no antiholes in the connectivity graph.

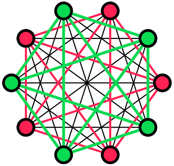
G



Lemma

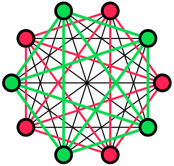
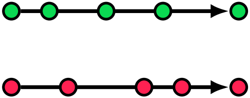
There are no antiholes in the connectivity graph.

G



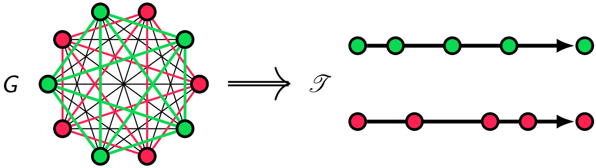
Lemma

There are no antiholes in the connectivity graph.

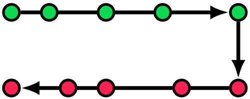
G  $\implies \mathscr{T}$ 

Lemma

There are no antiholes in the connectivity graph.

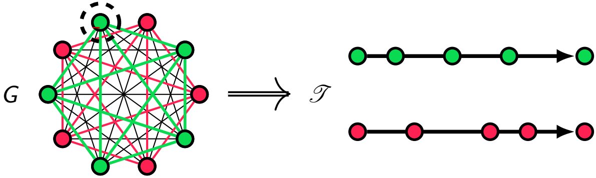


Case 1

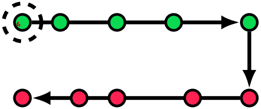


Lemma

There are no antiholes in the connectivity graph.

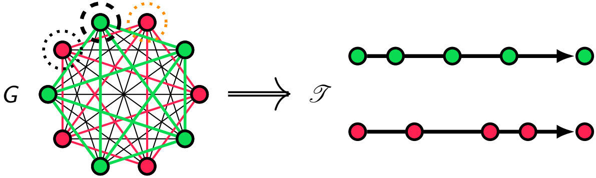


Case 1

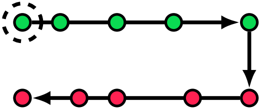


Lemma

There are no antiholes in the connectivity graph.

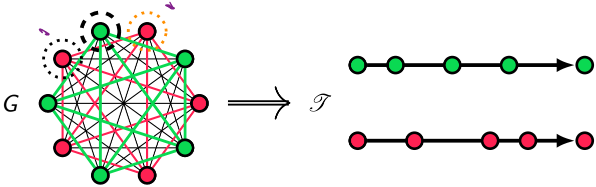


Case 1

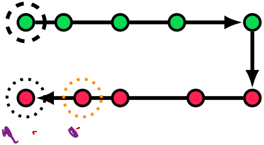


Lemma

There are no antiholes in the connectivity graph.

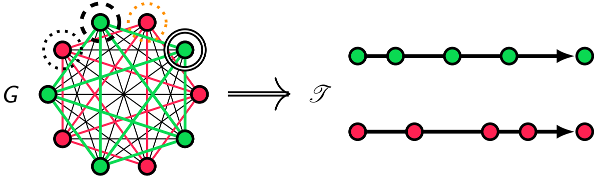


Case 1

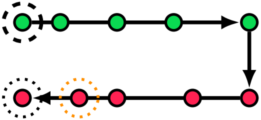


Lemma

There are no antiholes in the connectivity graph.

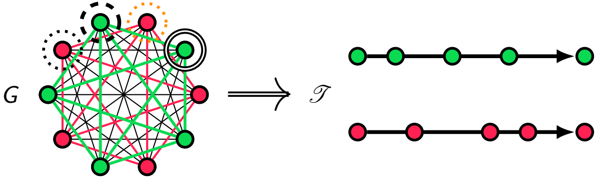


Case 1

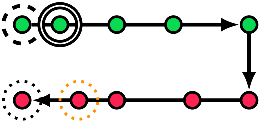


Lemma

There are no antiholes in the connectivity graph.

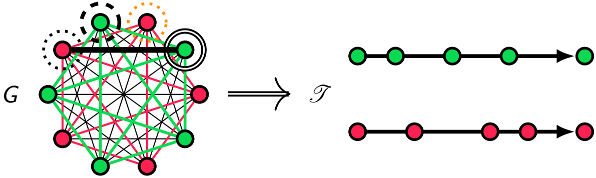


Case 1

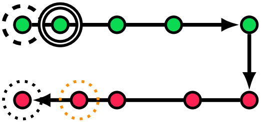


Lemma

There are no antiholes in the connectivity graph.



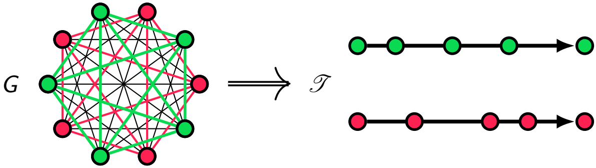
Case 1



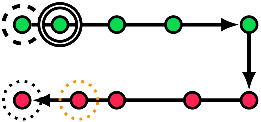
$\Rightarrow$ The temporal path of this edge does not exist in $\mathscr{T}$.

Lemma

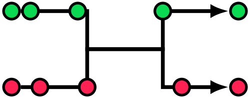
There are no antiholes in the connectivity graph.



Case 1



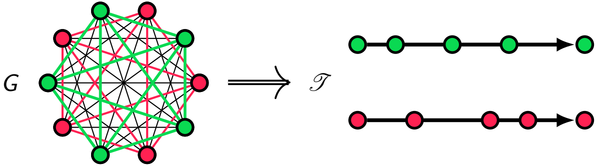
Case 2



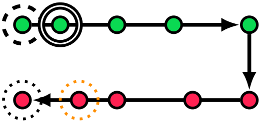
$\Rightarrow$ The temporal path of this edge does not exist in $\mathscr{T}$.

Lemma

There are no antiholes in the connectivity graph.

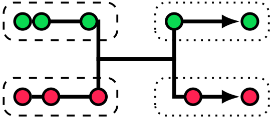


Case 1



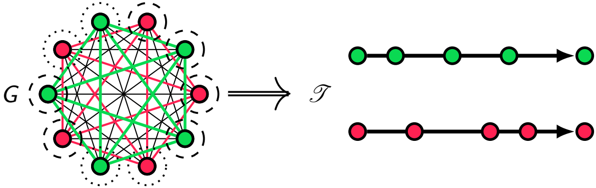
$\Rightarrow$ The temporal path of this edge does not exist in $\mathscr{T}$.

Case 2

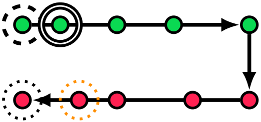


Lemma

There are no antiholes in the connectivity graph.

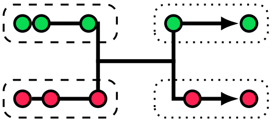


Case 1



⇒ The temporal path of this edge does not exist in $\mathscr{T}$.

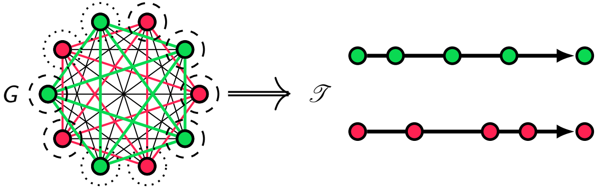
Case 2



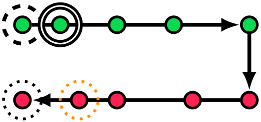
⇒ Each has to be complete bipartite in G

Lemma

There are no antiholes in the connectivity graph.

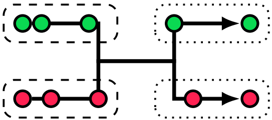


Case 1



$\Rightarrow$ The temporal path of this edge does not exist in $\mathscr{T}$.

Case 2



$\Rightarrow$ Each has to be complete bipartite in G, only possible if order ≤ 7, which we then manage.

Path Cover of temporal oriented trees (4) Conclusion

Lemmas

The connectivity graph is (hole,antihole)-free

Lemmas

The connectivity graph is (hole,antihole)-free $\Rightarrow$ It is **weakly chordal** (subclass of perfect)

Lemmas

The connectivity graph is (hole,antihole)-free $\Rightarrow$ It is **weakly chordal** (subclass of perfect) $\Rightarrow$ Dilworth property!

Lemmas

The connectivity graph is (hole,antihole)-free $\Rightarrow$ It is **weakly chordal** (subclass of perfect) $\Rightarrow$ Dilworth property!

Theorem [Hayward, Spinrad & Sritharan, 2000]

There is a $\mathcal{O}(mn)$ algorithm for Clique Cover in weakly chordal graphs with n vertices and m edges.

Lemmas

The connectivity graph is (hole,antihole)-free $\Rightarrow$ It is **weakly chordal** (subclass of perfect) $\Rightarrow$ Dilworth property!

Theorem [Hayward, Spinrad & Sritharan, 2000]

There is a $\mathcal{O}(mn)$ algorithm for Clique Cover in weakly chordal graphs with n vertices and m edges.

$\Rightarrow$ Connectivity graph in $\mathcal{O}(n^2\ell)$, then [HSS00] in $\mathcal{O}(n^2 \times n)$.

Lemmas

The connectivity graph is (hole,antihole)-free $\Rightarrow$ It is **weakly chordal** (subclass of perfect) $\Rightarrow$ Dilworth property!

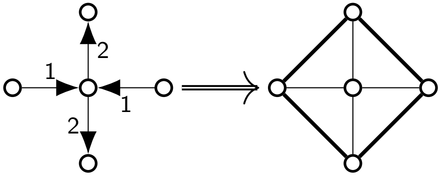
Theorem [Hayward, Spinrad & Sritharan, 2000]

There is a $\mathcal{O}(mn)$ algorithm for Clique Cover in weakly chordal graphs with n vertices and m edges.

$\Rightarrow$ Connectivity graph in $\mathcal{O}(n^2\ell)$, then [HSS00] in $\mathcal{O}(n^2 \times n)$.

Remark

The connectivity graph is not chordal (it may contain C_4).



Hardness results (1)

Hardness results (1)

Theorem [CDFK, 2024+]

TEMPORAL PATH COVER is NP-hard on Temporal DAGs.

Reduction from 3-DIMENSIONAL MATCHING (inspired by [Monnot and Toulous, 07])

Theorem [CDFK, 2024+]

TEMPORAL PATH COVER is NP-hard on Temporal DAGs.

Reduction from 3-DIMENSIONAL MATCHING (inspired by [Monnot and Toulous, 07])

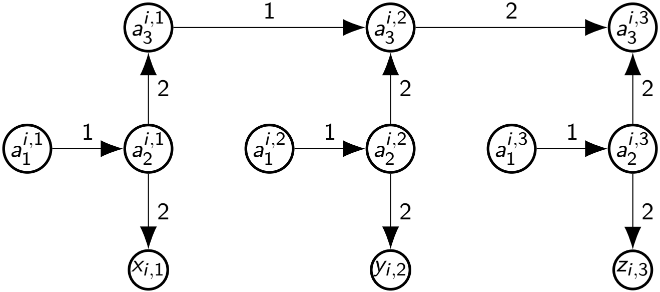


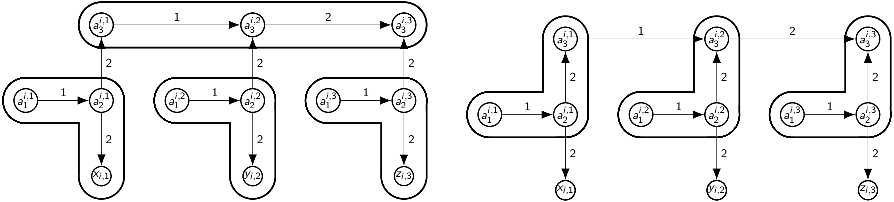
Figure: The gadget $H(s_i)$ for each triplet s_i.

Hardness results (1)

Theorem [CDFK, 2024+]

TEMPORAL PATH COVER is NP-hard on Temporal DAGs.

Reduction from 3-DIMENSIONAL MATCHING (inspired by [Monnot and Toulous, 07])



(a) $s_i \in M$ (b) $s_i \notin M$

Figure: Vertex partition of the gadget $H(s_i)$ into length-2 paths.

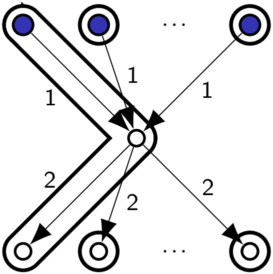
Total number of timesteps $= 2$ (bounded); Treewidth is unbounded; There are Temporal DAGs (transitive tournaments) without the Dilworth Property.

Hardness results (2)

Theorem [CDFK, 2024+]

Temporal Disjoint Path Cover is NP-hard on temporal oriented trees.

- Reduction from Unary Bin Packing (inspired by [Kunz, Molter, Zehavi, 23]);
- Treewidth = 1; Total number of timesteps = unbounded;
- Does not have TD-Dilworth property.



Parameterized complexity for TEMPORAL DISJOINT PATH COVER

Theorem [CDFK, 2024+]

TD-PC is FPT w.r.t. tw and $t_{\max}$ (total number of time-steps)

Dynamic programing on a nice tree decomposition

Theorem [CDFK, 2024+]

TD-PC is FPT w.r.t. tw and $t_{\max}$ (total number of time-steps)

Dynamic programing on a nice tree decomposition

Observation

Any arc of $\mathscr{D}$ appears in at most $t_{\max}$ paths of a TD-PC $\Rightarrow$ At most $p = \binom{\text{tw}}{2} \cdot t_{\max}$ temporally disjoint paths contain at least one arc from a given bag

Theorem [CDFK, 2024+]

TD-PC is FPT w.r.t. tw and $t_{\max}$ (total number of time-steps)

Dynamic programing on a nice tree decomposition

Observation

Any arc of $\mathscr{D}$ appears in at most $t_{\max}$ paths of a TD-PC $\Rightarrow$ At most $p = \binom{\mathrm{tw}}{2} \cdot t_{\max}$ temporally disjoint paths contain at least one arc from a given bag

For simplicity, duplicate the arcs such that each has only one time label (so a TD-PC uses arc-disjoint paths)

Parameterized complexity for TD-PC (2) Type

Type: necessary information at each node v

$\Rightarrow$ At most types for any node

Parameterized complexity for TD-PC (2) Type

Type: necessary information at each node v

- A partition $Q_0, Q_1, \ldots, Q_t$ of the arcs inside X_v (Q_i for $i \neq 0$ is in a temporal path P_i of a TD-PC, Q_0 is the unused arcs)

$\Rightarrow$ At most p^p types for any node

Parameterized complexity for TD-PC (2) Type

Type: necessary information at each node v

- A partition $Q_0, Q_1, \ldots, Q_t$ of the arcs inside X_v (Q_i for $i \neq 0$ is in a temporal path P_i of a TD-PC, Q_0 is the unused arcs)
- For each Q_i, the vertices V_i of X_v that are in P_i (endpoints of arcs in Q_i and those not incident with arcs in Q_i)

$\Rightarrow$ At most $p^p \times 2^{\mathrm{tw}+1}$ types for any node

Parameterized complexity for TD-PC (2) Type

Type: necessary information at each node v

- A partition $Q_0, Q_1, \ldots, Q_t$ of the arcs inside X_v (Q_i for $i \neq 0$ is in a temporal path P_i of a TD-PC, Q_0 is the unused arcs)
- For each Q_i, the vertices V_i of X_v that are in P_i (endpoints of arcs in Q_i and those not incident with arcs in Q_i)
- For each V_i, their order of occupation by P_i

$\Rightarrow$ At most $p^p \times 2^{\mathrm{tw}+1} \times (\mathrm{tw}+1)!$ types for any node

Parameterized complexity for TD-PC (2) Type

Type: necessary information at each node v

- A partition $Q_0, Q_1, \ldots, Q_t$ of the arcs inside X_v (Q_i for $i \neq 0$ is in a temporal path P_i of a TD-PC, Q_0 is the unused arcs)
- For each Q_i, the vertices V_i of X_v that are in P_i (endpoints of arcs in Q_i and those not incident with arcs in Q_i)
- For each V_i, their order of occupation by P_i
- For each Q_i, the vertices in V_i with one or two arcs outside of X_v, the time labels of those arcs, and whether the neighbour appears below or above v in the decomposition

$\Rightarrow$ At most $p^p \times 2^{\mathrm{tw}+1} \times (\mathrm{tw}+1)! \times 2^{\mathrm{tw}+2} \times t_{\max}^2$ types for any node

Type: necessary information at each node v

- A partition $Q_0, Q_1, \dots, Q_t$ of the arcs inside X_v (Q_i for $i \neq 0$ is in a temporal path P_i of a TD-PC, Q_0 is the unused arcs)
- For each Q_i, the vertices V_i of X_v that are in P_i (endpoints of arcs in Q_i and those not incident with arcs in Q_i)
- For each V_i, their order of occupation by P_i
- For each Q_i, the vertices in V_i with one or two arcs outside of X_v, the time labels of those arcs, and whether the neighbour appears below or above v in the decomposition

$\Rightarrow$ At most $p^p \times 2^{\mathrm{tw}+1} \times (\mathrm{tw}+1)! \times 2^{\mathrm{tw}+2} \times t_{\max}^2 \in 2^{\mathcal{O}(p \log p)}$ types for any node

Parameterized complexity for TD-PC (3) Consistency

Consistency of a type

- The ordered vertices V_i, the arcs of Q_i, and the information about the arcs going outside of X_v, induce temporal paths
- The arcs going outside of X_v exist in the digraph and their labels are compatible with the order
- Every vertex of X_v is in a V_i

Now, we compute from the bottom-up, maintaining consistency.

Dynamic programing using consistent types of partial solutions

- **Leaf node:** No partial solution since empty
- **Introduce node:** Check compatibility with the child (either *a* is in a path in the type, or *a* is added as a single-vertex path)
- **Forget node:** Check compatibility with child (the types are the ones obtained by removing the vertex *a*), discard those where *a* has an arc going above
- **Join node:** Check compatibility of the children (partition of arcs, order of vertices, neighbours outside of the bag, are they above or below in the decomposition, ... $\Rightarrow$ all have to agree), don't count twice the paths that intersect the bag

Running time $2^{\mathcal{O}(p \log p)} n$, so FPT w.r.t. $p = f(\mathrm{tw}, t_{\max})$

Dynamic programing using consistent types of partial solutions

- **Leaf node:** No partial solution since empty
- **Introduce node:** Check compatibility with the child (either a is in a path in the type, or a is added as a single-vertex path)
- **Forget node:** Check compatibility with child (the types are the ones obtained by removing the vertex a), discard those where a has an arc going above
- **Join node:** Check compatibility of the children (partition of arcs, order of vertices, neighbours outside of the bag, are they above or below in the decomposition, ... $\Rightarrow$ all have to agree), don't count twice the paths that intersect the bag

Running time $2^{\mathcal{O}(p \log p)} n$, so FPT w.r.t. $p = f(\mathrm{tw}, t_{\max})$

And for TPC?

Same principle, but the paths can intersect

Parameterized complexity for TD-PC (4) Computation

Dynamic programing using consistent types of partial solutions

- **Leaf node:** No partial solution since empty
- **Introduce node:** Check compatibility with the child (either a is in a path in the type, or a is added as a single-vertex path)
- **Forget node:** Check compatibility with child (the types are the ones obtained by removing the vertex a), discard those where a has an arc going above
- **Join node:** Check compatibility of the children (partition of arcs, order of vertices, neighbours outside of the bag, are they above or below in the decomposition, ... $\Rightarrow$ all have to agree), don't count twice the paths that intersect the bag

Running time $2^{\mathcal{O}(p \log p)} n$, so FPT w.r.t. $p = f(\mathrm{tw}, t_{\max})$

And for TPC?

Same principle, but the paths can intersect $\Rightarrow$ More information in type: how many times in the solution does Q_i appear

Dynamic programing using consistent types of partial solutions

- **Leaf node:** No partial solution since empty
- **Introduce node:** Check compatibility with the child (either a is in a path in the type, or a is added as a single-vertex path)
- **Forget node:** Check compatibility with child (the types are the ones obtained by removing the vertex a), discard those where a has an arc going above
- **Join node:** Check compatibility of the children (partition of arcs, order of vertices, neighbours outside of the bag, are they above or below in the decomposition, ... $\Rightarrow$ all have to agree), don't count twice the paths that intersect the bag

Running time $2^{\mathcal{O}(p \log p)} n$, so FPT w.r.t. $p = f(\mathrm{tw}, t_{\max})$

And for TPC?

Same principle, but the paths can intersect $\Rightarrow$ More information in type: how many times in the solution does Q_i appear $\Rightarrow$ Running time $k^{\mathcal{O}(p \log p)} n$ where $k \in \mathcal{O}(n)$ is the solution size $\Rightarrow$ XP w.r.t. p

Conclusion and future work

Temporal class	TPC	TD-PC
Oriented paths	$\mathcal{O}(\ell n)$	$\mathcal{O}(\ell n)$
Rooted trees	$\mathcal{O}(\ell n^2)$	$\mathcal{O}(\ell n^2)$
Oriented trees	$\mathcal{O}(\ell n^2 + n^3)$	NP-hard
DAGs*	NP-hard	NP-hard
Digraphs	XP (tw and $t_{\max}$) $n^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}$	FPT (tw and $t_{\max}$) $2^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}\, n$

Conclusion and future work

Temporal class	TPC	TD-PC
Oriented paths	$\mathcal{O}(\ell n)$	$\mathcal{O}(\ell n)$
Rooted trees	$\mathcal{O}(\ell n^2)$	$\mathcal{O}(\ell n^2)$
Oriented trees	$\mathcal{O}(\ell n^2 + n^3)$	NP-hard
DAGs*	NP-hard	NP-hard
Digraphs	XP (tw and $t_{\max}$) $n^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}$	FPT (tw and $t_{\max}$) $2^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}\, n$

Perspectives

- Better FPT, FPT for TPC?
- Approximation?
- Classes of oriented trees where TD-PC is polynomial?
- Other temporal problems that can be reduced to a static problem?

Conclusion and future work

Temporal class	TPC	TD-PC
Oriented paths	$\mathcal{O}(\ell n)$	$\mathcal{O}(\ell n)$
Rooted trees	$\mathcal{O}(\ell n^2)$	$\mathcal{O}(\ell n^2)$
Oriented trees	$\mathcal{O}(\ell n^2 + n^3)$	NP-hard
DAGs*	NP-hard	NP-hard
Digraphs	XP (tw and $t_{\max}$) $n^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}$	FPT (tw and $t_{\max}$) $2^{\mathcal{O}(\mathrm{tw}^2\, t_{\max} \log(\mathrm{tw}\, t_{\max}))}\, n$

Perspectives

- Better FPT, FPT for TPC?
- Approximation?
- Classes of oriented trees where TD-PC is polynomial?
- Other temporal problems that can be reduced to a static problem?