Counting Temporal Paths

Jessica Enright[1] Kitty Meeks[1] Hendrik Molter[2]

[1] School of Computing Science, University of Glasgow, Glasgow, UK
[2] Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Israel

Algorithmic Aspects of Temporal Graphs V

Project initiated at Dagstuhl Seminar "Temporal Graphs: Structure, Algorithms, Applications".

Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

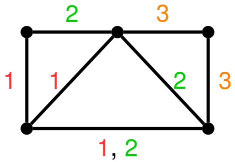
A **temporal graph** $\mathscr{G} = (V, (E_i)_{i \in [T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathscr{G}$.

Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

A **temporal graph** $\mathcal{G} = (V, (E_i)_{i \in [T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.

$\mathcal{G}$:

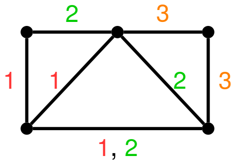


Temporal Graphs and Temporal Paths: Notation and Definition

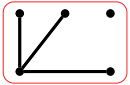
Temporal Graph

A **temporal graph** $\mathcal{G} = (V, (E_i)_{i \in [T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.

$\mathcal{G}$:



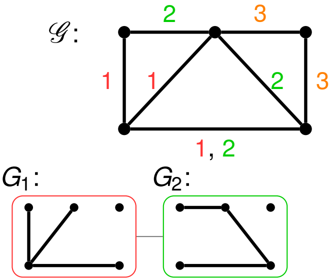
G_1:



Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

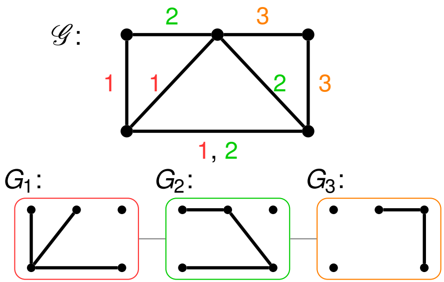
A **temporal graph** $\mathcal{G} = (V, (E_i)_{i \in [T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

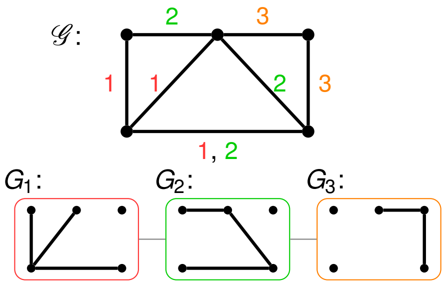
A **temporal graph** $\mathcal{G} = (V, (E_i)_{i\in[T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

A **temporal graph** $\mathcal{G} = (V, (E_i)_{i\in[T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal (s, z)-Path

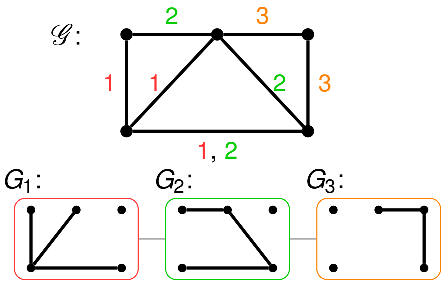
Sequence of time edges forming a path from s to z that have:

- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).

Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

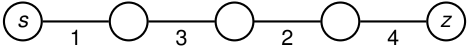
A **temporal graph** $\mathcal{G} = (V, (E_i)_{i\in[T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).

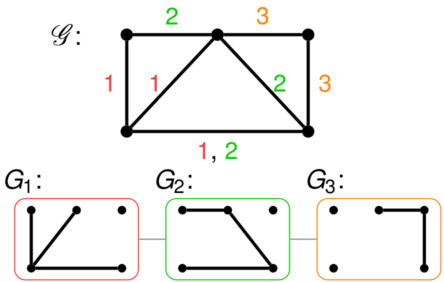


Not a temporal path.

Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

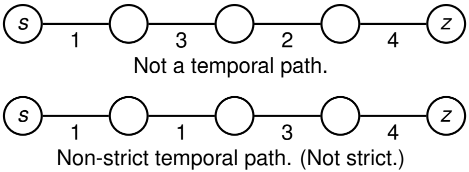
A **temporal graph** $\mathcal{G} = (V, (E_i)_{i\in[T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

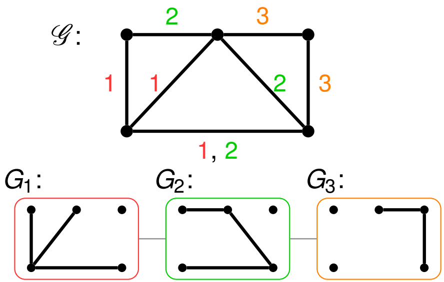
- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).



Temporal Graphs and Temporal Paths: Notation and Definition

Temporal Graph

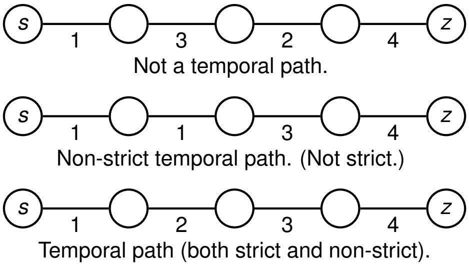
A **temporal graph** $\mathcal{G} = (V, (E_i)_{i \in [T]})$ is a vertex set V with a list of edge sets $E_1, \ldots, E_T$ over V, where T is the lifetime of $\mathcal{G}$.



Temporal (s, z)-Path

Sequence of time edges forming a path from s to z that have:

- increasing time stamps (strict).
- non-decreasing time stamps (non-strict).



Motivation: Temporal Betweenness

"How important is Berlin main station as a hub for the public transportation network?"

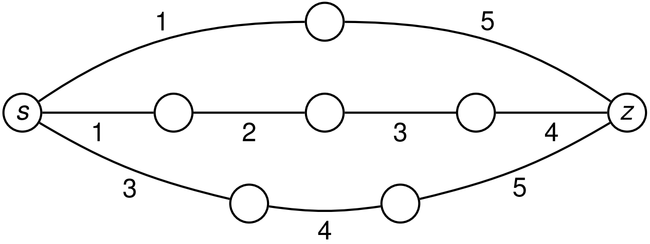
Motivation: Temporal Betweenness

"How important is Berlin main station as a hub for the public transportation network?"

Betweenness of a vertex v in a graph $G = (V, E)$:
"How likely is a shortest (optimal) path to pass through vertex v?"

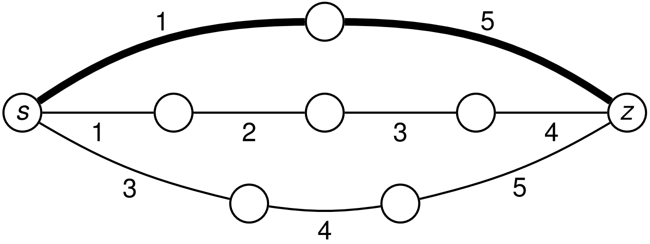
Optimal Temporal Paths

Which temporal path from s to z is optimal?



Optimal Temporal Paths

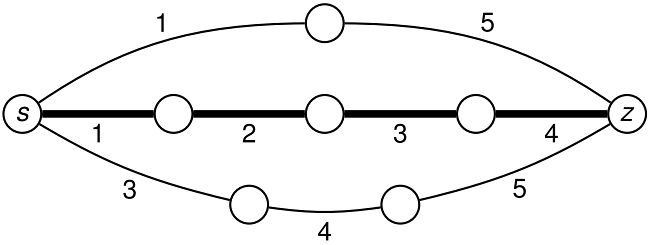
Which temporal path from s to z is optimal?



- **Shortest** temporal paths use the minimum number of edges.

Optimal Temporal Paths

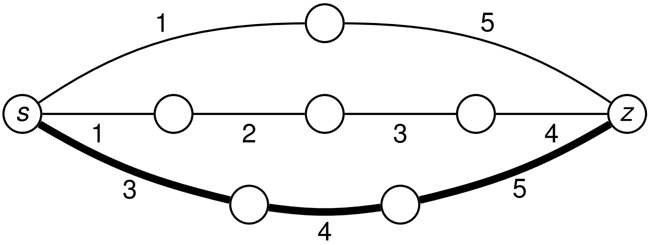
Which temporal path from s to z is optimal?



- **Shortest** temporal paths use the minimum number of edges.
- **Foremost** temporal paths have a minimum arrival time.

Optimal Temporal Paths

Which temporal path from s to z is optimal?



- **Shortest** temporal paths use the minimum number of edges.
- **Foremost** temporal paths have a minimum arrival time.
- **Fastest** temporal paths have a minimum difference between starting and arrival time.

Temporal Betweenness: Definition

Temporal Betweenness Centrality

$$C_B^{(\star)}(v) = \sum_{s \neq v \neq z} \begin{cases} 0 & \nexists \text{ temp.}(s,z)\text{-path} \\ \frac{\sigma_{sz}^{(\star)}(v)}{\sigma_{sz}^{(\star)}} & \text{otherwise} \end{cases}$$

$\sigma_{sz}^{(\star)}$: # $\star$-temp. paths from s to z

$\sigma_{sz}^{(\star)}(v)$: # $\star$-temp. paths from s to z via v

Temporal Betweenness: Definition

Temporal Betweenness Centrality

$$C_B^{(\star)}(v) = \sum_{s \neq v \neq z} \begin{cases} 0 & \nexists \text{ temp.}(s,z)\text{-path} \\ \frac{\sigma_{sz}^{(\star)}(v)}{\sigma_{sz}^{(\star)}} & \text{otherwise} \end{cases}$$

$\sigma_{sz}^{(\star)}$: # $\star$-temp. paths from s to z

$\sigma_{sz}^{(\star)}(v)$: # $\star$-temp. paths from s to z via v

$\star$ denotes "optimality concept"

Temporal Betweenness: Definition

Temporal Betweenness Centrality

$$C_B^{(\star)}(v) = \sum_{s \neq v \neq z} \begin{cases} 0 & \nexists \text{ temp.}(s,z)\text{-path} \\ \frac{\sigma_{sz}^{(\star)}(v)}{\sigma_{sz}^{(\star)}} & \text{otherwise} \end{cases}$$

$\sigma_{sz}^{(\star)}$: # $\star$-temp. paths from s to z $\sigma_{sz}^{(\star)}(v)$: # $\star$-temp. paths from s to z via v

$\star$ denotes “optimality concept”

- Computing temporal betweenness essentially equivalent to counting (optimal) temporal paths.

Temporal Betweenness: Definition

Temporal Betweenness Centrality

$$C_B^{(\star)}(v) = \sum_{s \neq v \neq z} \begin{cases} 0 & \nexists \text{ temp.}(s,z)\text{-path} \\ \frac{\sigma_{sz}^{(\star)}(v)}{\sigma_{sz}^{(\star)}} & \text{otherwise} \end{cases}$$

$\sigma_{sz}^{(\star)}$: # $\star$-temp. paths from s to z $\qquad$ $\sigma_{sz}^{(\star)}(v)$: # $\star$-temp. paths from s to z via v

$\star$ denotes "optimality concept"

- Computing temporal betweenness essentially equivalent to counting (optimal) temporal paths.
- For many interesting optimality concepts such as "foremost" or "fastest" the corresponding counting problem is **#P-hard**.

Temporal Path Counting I

#Temporal Path

Input: A temporal graph $\mathcal{G} = (V, (E_i)_{i \in [T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathcal{G}$.

Temporal Path Counting I

#Temporal Path	
Input:	A temporal graph $\mathscr{G} = (V, (E_i)_{i\in[T]})$ and two vertices $s, z \in V$.
Task:	Count the temporal (s, z)-paths in $\mathscr{G}$.

Remarks:

- Solving **#Temporal Path** allows to count **foremost** temporal paths by removing all “late” time steps.

Temporal Path Counting I

#Temporal Path

Input: A temporal graph $\mathscr{G} = (V, (E_i)_{i \in [T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathscr{G}$.

Remarks:

- Solving **#Temporal Path** allows to count **foremost** temporal paths by removing all "late" time steps.
- Solving **#Temporal Path** allows to count **fastest** temporal paths with linear overhead (in T).

Temporal Path Counting I

#Temporal Path

Input: A temporal graph $\mathscr{G} = (V, (E_i)_{i \in [T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathscr{G}$.

Remarks:

- Solving **#Temporal Path** allows to count **foremost** temporal paths by removing all "late" time steps.
- Solving **#Temporal Path** allows to count **fastest** temporal paths with linear overhead (in T).
- One can count temporal (s, z)-paths via a vertex v by removing v from $\mathscr{G}$.

Temporal Path Counting I

#Temporal Path

Input: A temporal graph $\mathcal{G} = (V, (E_i)_{i \in [T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathcal{G}$.

Remarks:

- Solving **#Temporal Path** allows to count **foremost** temporal paths by removing all “late” time steps.
- Solving **#Temporal Path** allows to count **fastest** temporal paths with linear overhead (in T).
- One can count temporal (s, z)-paths via a vertex v by removing v from $\mathcal{G}$.
- For this talk: focus on **non-strict** temporal paths.

Temporal Path Counting II

#Temporal Path

Input: A temporal graph $\mathcal{G} = (V, (E_i)_{i\in[T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathcal{G}$.

Observation

#Temporal Path is #P-hard even if $T = 1$.

Temporal Path Counting II

#Temporal Path

Input: A temporal graph $\mathcal{G} = (V, (E_i)_{i\in[T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathcal{G}$.

Observation

#Temporal Path is #P-hard even if $T = 1$.

Parameterized Counting

Temporal Path Counting II

#Temporal Path

Input: A temporal graph $\mathcal{G} = (V, (E_i)_{i \in [T]})$ and two vertices $s, z \in V$.

Task: Count the temporal (s, z)-paths in $\mathcal{G}$.

Observation

#Temporal Path is #P-hard even if $T = 1$.

Parameterized Counting

Approximate Counting

Warm-Up: Temporal Path Counting on Forests

Observation

Underlying graph is forest $\Rightarrow$ same sequence of vertices visited by any temporal (s, z)-path.

Warm-Up: Temporal Path Counting on Forests

Observation

Underlying graph is forest $\Rightarrow$ same sequence of vertices visited by any temporal (s, z)-path.

Let $s = v_1, v_2, \ldots, v_\ell = z$ be the sequence of vertices visited by any temporal (s, z)-path.

Warm-Up: Temporal Path Counting on Forests

Observation

Underlying graph is forest $\Rightarrow$ same sequence of vertices visited by any temporal (s, z)-path.

Let $s = v_1, v_2, \ldots, v_\ell = z$ be the sequence of vertices visited by any temporal (s, z)-path.

Use dynamic program
$F(v, t) :=$ number of temporal (s, v)-paths that arrive in v at time t or earlier.

Warm-Up: Temporal Path Counting on Forests

Observation

Underlying graph is forest $\Rightarrow$ same sequence of vertices visited by any temporal (s, z)-path.

Let $s = v_1, v_2, \ldots, v_\ell = z$ be the sequence of vertices visited by any temporal (s, z)-path.

Use dynamic program
$F(v, t) :=$ number of temporal (s, v)-paths that arrive in v at time t or earlier.

$$F(v_1 = s, t) = 1$$
$$F(v_i, t) = \sum_{t' \leq t \text{ with } \{v_{i-1}, v_i\} \in E_{t'}} F(v_{i-1}, t') \text{ for } 1 < i \leq \ell.$$

Warm-Up: Temporal Path Counting on Forests

Observation

Underlying graph is forest $\Rightarrow$ same sequence of vertices visited by any temporal (s, z)-path.

Let $s = v_1, v_2, \ldots, v_\ell = z$ be the sequence of vertices visited by any temporal (s, z)-path.

Use dynamic program
$F(v, t) :=$ number of temporal (s, v)-paths that arrive in v at time t or earlier.

$$F(v_1 = s, t) = 1$$
$$F(v_i, t) = \sum_{t' \leq t \text{ with } \{v_{i-1}, v_i\} \in E_{t'}} F(v_{i-1}, t') \text{ for } 1 < i \leq \ell.$$

Theorem

#Temporal Path solvable in polynomial time if the underlying graph is a **forest**.

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Theorem

#Temporal Path is $\oplus$W[1]-hard when parameterized by the **feedback vertex number** of the underlying graph.

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Theorem

#Temporal Path is $\oplus$W[1]-hard when parameterized by the **feedback vertex number** of the underlying graph.

Main Ideas:

- Jiang [TCS 2010] showed that **Independent Set on 2-Track Interval Graphs** is W[1]-hard when parameterized by the solution size.

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Theorem

#Temporal Path is $\oplus$W[1]-hard when parameterized by the **feedback vertex number** of the underlying graph.

Main Ideas:

- Jiang [TCS 2010] showed that **Independent Set on 2-Track Interval Graphs** is W[1]-hard when parameterized by the solution size.

  Each vertex v represented by two intervals a_v, b_v.

  Vertices v, w have an edge iff $a_v \cap a_w \neq \emptyset$ or $b_v \cap b_w \neq \emptyset$.

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Theorem

#Temporal Path is $\oplus$W[1]-hard when parameterized by the **feedback vertex number** of the underlying graph.

Main Ideas:

- Jiang [TCS 2010] showed that **Independent Set on 2-Track Interval Graphs** is W[1]-hard when parameterized by the solution size.

  Each vertex v represented by two intervals a_v, b_v.

  Vertices v, w have an edge iff $a_v \cap a_w \neq \emptyset$ or $b_v \cap b_w \neq \emptyset$.
- Investigating the reduction shows that $\oplus$**Multicolored Independent Set on 2-Track Interval Graphs** is $\oplus$W[1]-hard when parameterized by the number of colors.

Parameterized Hardness I

Question: How can we generalize the forest algorithm?

Theorem

#Temporal Path is $\oplus$W[1]-hard when parameterized by the **feedback vertex number** of the underlying graph.

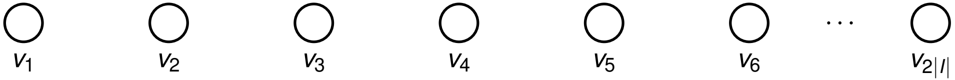
Main Ideas:

- Jiang [TCS 2010] showed that **Independent Set on 2-Track Interval Graphs** is W[1]-hard when parameterized by the solution size.

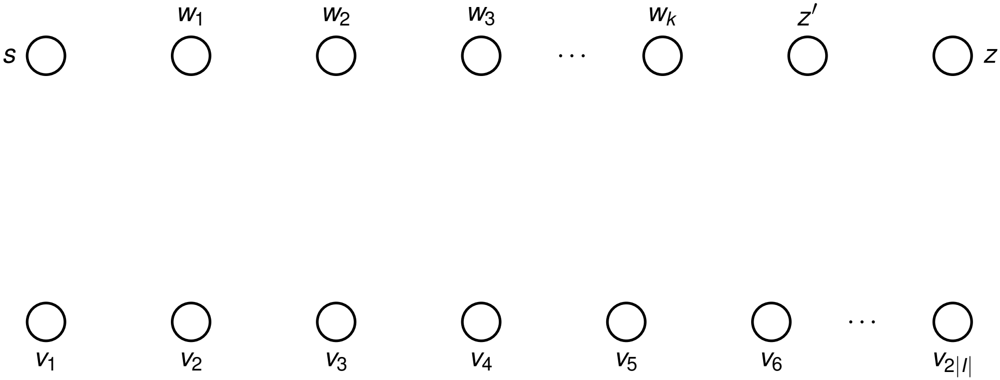
  Each vertex v represented by two intervals a_v, b_v.

  Vertices v, w have an edge iff $a_v \cap a_w \neq \emptyset$ or $b_v \cap b_w \neq \emptyset$.
- Investigating the reduction shows that $\oplus$**Multicolored Independent Set on 2-Track Interval Graphs** is $\oplus$W[1]-hard when parameterized by the number of colors.
- Model one track with vertices and one with time.

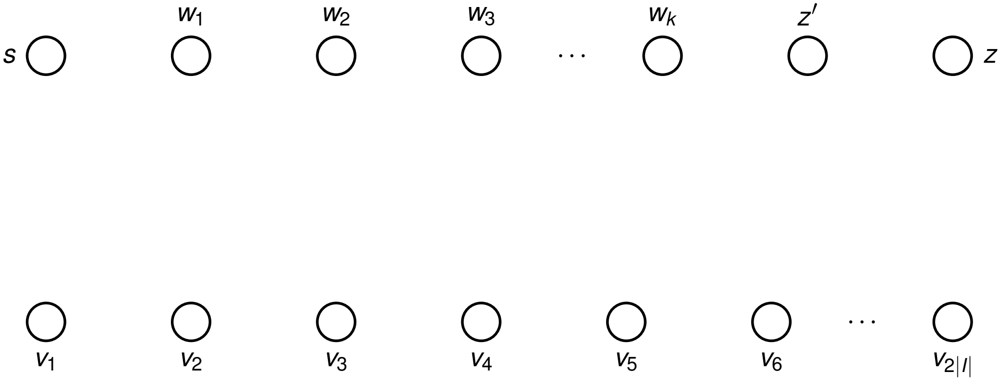
Parameterized Hardness II



Parameterized Hardness II

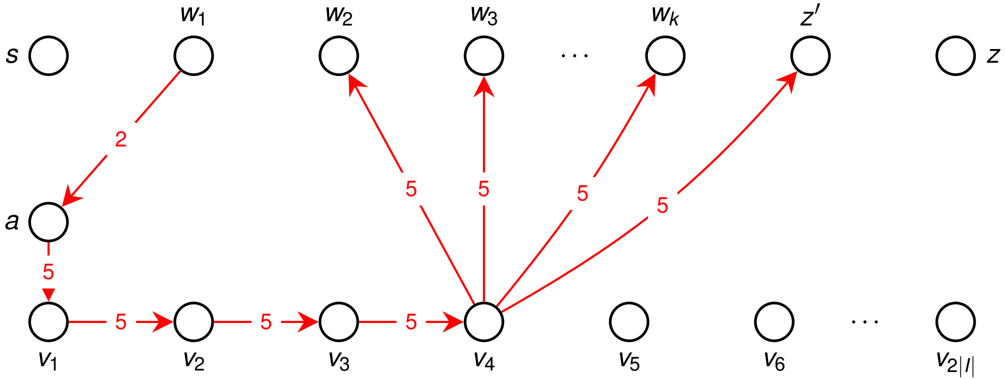


Parameterized Hardness II



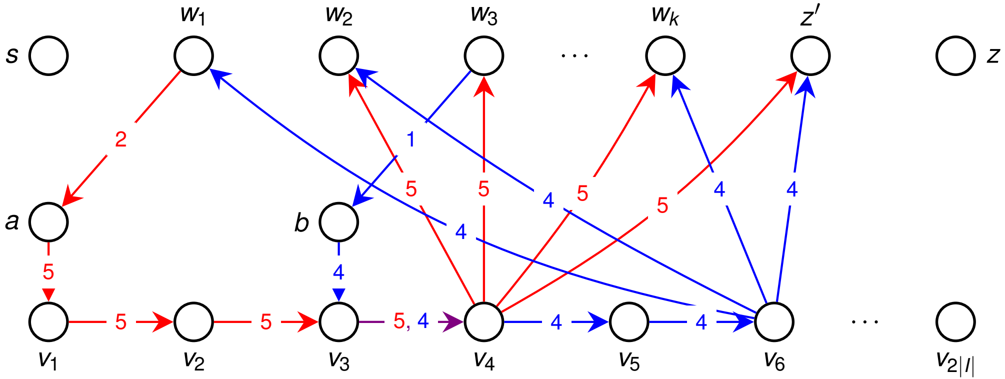
Two interval pairs $a = ([1,4],[2,5])$ and $b = ([3,6],[1,4])$, where $c(a) = 1$, $c(b) = 3$.

Parameterized Hardness II



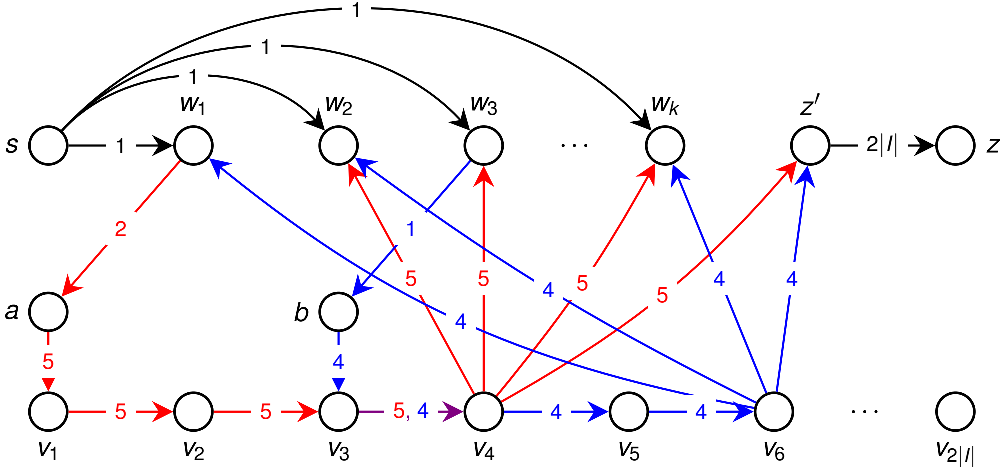
Two interval pairs $a = ([1,4],[2,5])$ and $b = ([3,6],[1,4])$, where $c(a) = 1$, $c(b) = 3$.

Parameterized Hardness II



Two interval pairs $a = ([1,4],[2,5])$ and $b = ([3,6],[1,4])$, where $c(a) = 1$, $c(b) = 3$.

Parameterized Hardness II



Two interval pairs $a = ([1,4],[2,5])$ and $b = ([3,6],[1,4])$, where $c(a) = 1$, $c(b) = 3$.

Parameterized Hardness III

Problem: “Cheating” temporal paths.

Parameterized Hardness III

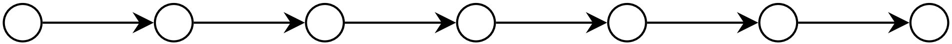
Problem: “Cheating” temporal paths.

Idea: There should always be an even number of cheating temporal (s, z)-paths.

Parameterized Hardness III

Problem: “Cheating” temporal paths.

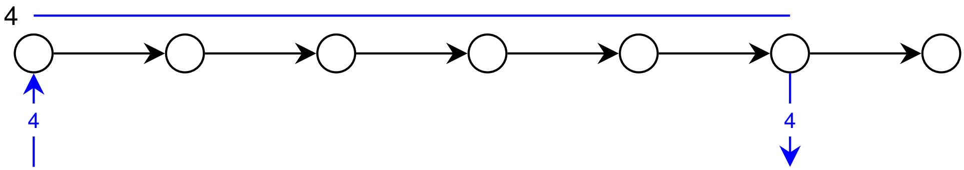
Idea: There should always be an even number of cheating temporal (s, z)-paths.



Parameterized Hardness III

Problem: “Cheating” temporal paths.

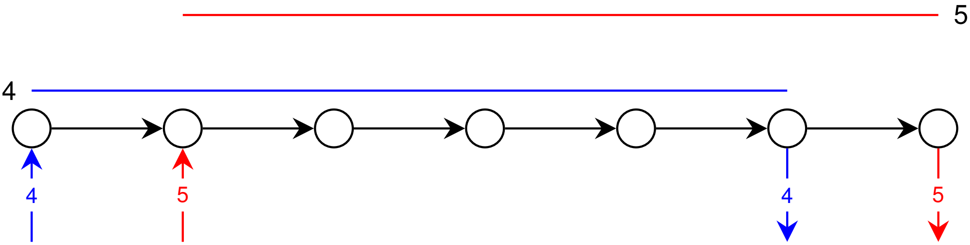
Idea: There should always be an even number of cheating temporal (s, z)-paths.



Parameterized Hardness III

Problem: "Cheating" temporal paths.

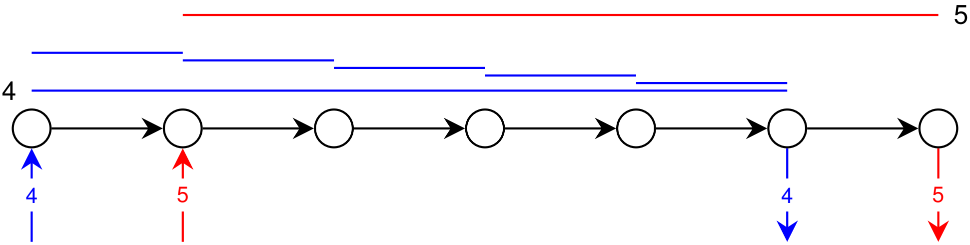
Idea: There should always be an even number of cheating temporal (s, z)-paths.



Parameterized Hardness III

Problem: "Cheating" temporal paths.

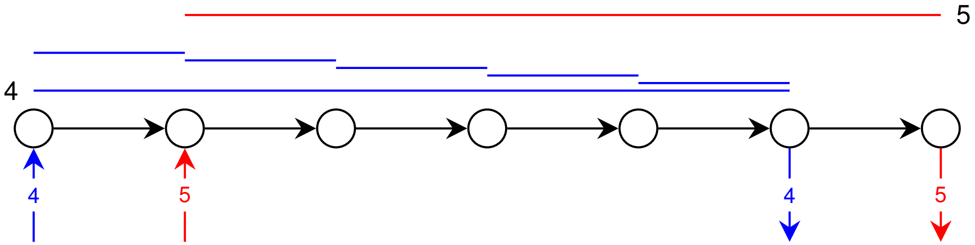
Idea: There should always be an even number of cheating temporal (s, z)-paths.



Parameterized Hardness III

Problem: “Cheating” temporal paths.

Idea: There should always be an even number of cheating temporal (s, z)-paths.



Lemma

The parity of temporal (s, z)-paths equals the parity of **colorful** independent sets.

Generalizations of the Forest Algorithm and Further Results

Theorem

#Temporal Path is in FPT when parameterized by the **feedback edge number** of the underlying graph.

Generalizations of the Forest Algorithm and Further Results

Theorem

#Temporal Path is in FPT when parameterized by the **feedback edge number** of the underlying graph.

Theorem

#Temporal Path is in FPT when parameterized by the **timed feedback vertex number**.

Generalizations of the Forest Algorithm and Further Results

Theorem

#Temporal Path is in FPT when parameterized by the **feedback edge number** of the underlying graph.

Theorem

#Temporal Path is in FPT when parameterized by the **timed feedback vertex number**.

Theorem

#Temporal Path is in FPT when parameterized by the **treewidth** of the underlying graph and the **lifetime** T.

Theorem

#Temporal Path is in FPT when parameterized by the **vertex-interval-membership-width**.

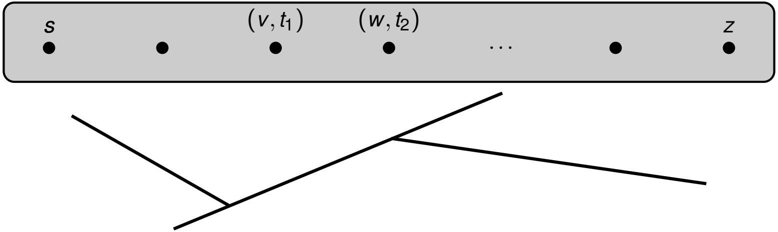
Timed Feedback Vertex Number I

Timed Feedback Vertex Set

Minimum size $X \subseteq V \times [T]$ such that underlying graph of $\mathcal{G} - X$ is a forest.

TFVS:

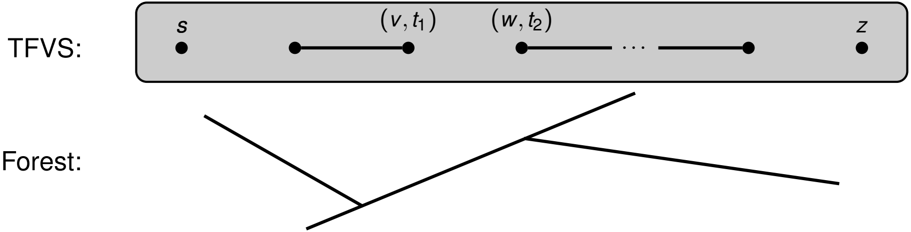
Forest:



Timed Feedback Vertex Number I

Timed Feedback Vertex Set

Minimum size $X \subseteq V \times [T]$ such that underlying graph of $\mathcal{G} - X$ is a forest.

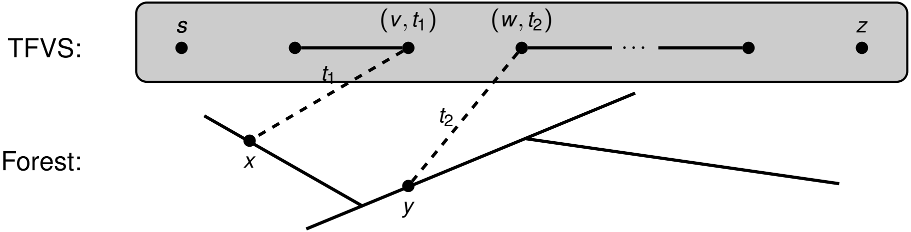


- Go through all variants which (and in which order) to traverse TFVS appearances.

Timed Feedback Vertex Number I

Timed Feedback Vertex Set

Minimum size $X \subseteq V \times [T]$ such that underlying graph of $\mathcal{G} - X$ is a forest.

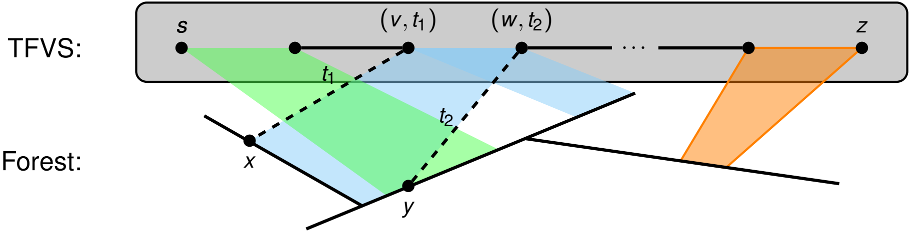


- Go through all variants which (and in which order) to traverse TFVS appearances.

Timed Feedback Vertex Number I

Timed Feedback Vertex Set

Minimum size $X \subseteq V \times [T]$ such that underlying graph of $\mathcal{G} - X$ is a forest.

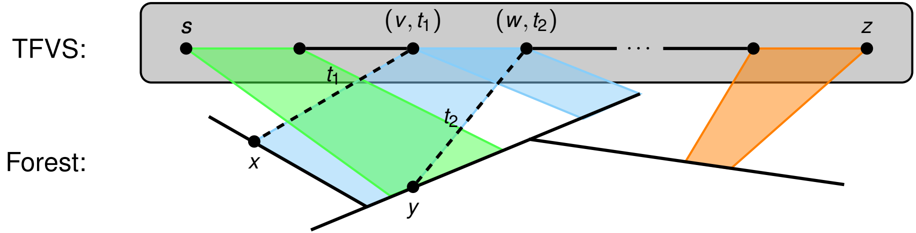


- Go through all variants which (and in which order) to traverse TFVS appearances.
- Use forest algorithm to compute all ways to close "gaps" between TFVS appearances.

Timed Feedback Vertex Number I

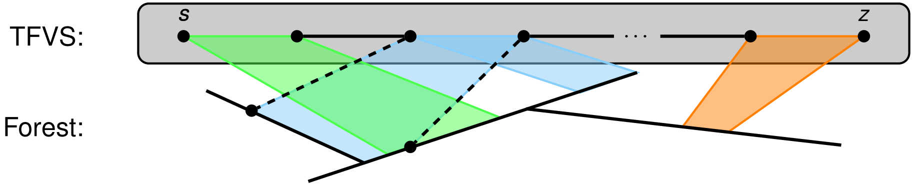
Timed Feedback Vertex Set

Minimum size $X \subseteq V \times [T]$ such that underlying graph of $\mathcal{G} - X$ is a forest.

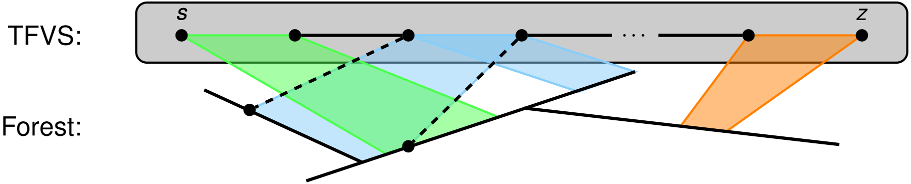


- Go through all variants which (and in which order) to traverse TFVS appearances.
- Use forest algorithm to compute all ways to close "gaps" between TFVS appearances.
- For every valid way to glue together a temporal (s, z)-path, the number of temporal (s, z)-paths visiting the same vertex sequence equals the **product** of the number of ways to close each gap.

Timed Feedback Vertex Number II

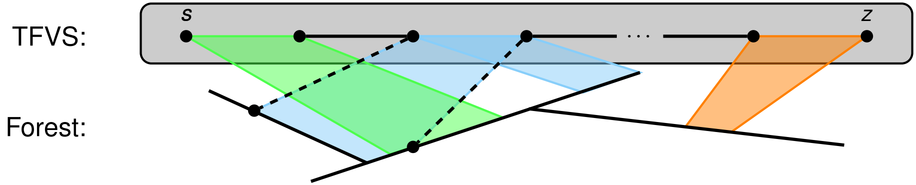


Timed Feedback Vertex Number II



Note: Intersection graphs of paths segments in forests are **chordal**.

Timed Feedback Vertex Number II



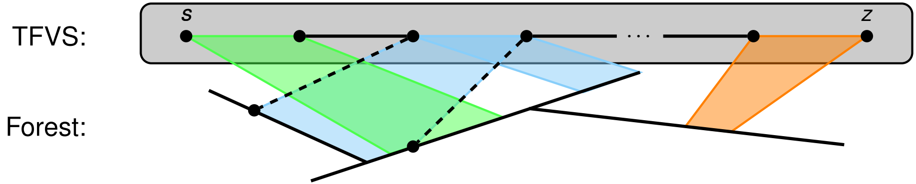
Note: Intersection graphs of paths segments in forests are **chordal**.

#Weighted Multicolored Independent Set on Chordal Graphs

Input: A chordal graph $G = (V, E)$, a coloring $c : V \to [k]$, and weights $w : V \to \mathbb{N}$.

Task: Compute $\sum_{X \subseteq V \mid X \text{ is a multicolored independent set in } G} \prod_{v \in X} w(v)$.

Timed Feedback Vertex Number II



Note: Intersection graphs of paths segments in forests are **chordal**.

#Weighted Multicolored Independent Set on Chordal Graphs	
Input:	A chordal graph $G = (V, E)$, a coloring $c : V \to [k]$, and weights $w : V \to \mathbb{N}$.
Task:	Compute $\sum_{X \subseteq V \mid X \text{ is a multicolored independent set in } G} \prod_{v \in X} w(v)$.

Idea: Adapt FPT algorithm for **Multicolored Independent Set on Chordal Graphs** parameterized by number of colors by Bentert, van Bevern, and Niedermeier [JOSH 2019].

Approximation Results

Theorem

There is no FPRAS for **#Temporal Path** unless NP$=$BPP.

Approximation Results

Theorem

There is no FPRAS for **#Temporal Path** unless NP$=$BPP.

Theorem

#Temporal Path admits an FPTRAS when parameterized by the maximum length of a temporal (s, z)-path.

Approximation Results

Theorem

There is no FPRAS for **#Temporal Path** unless NP=BPP.

Theorem

#Temporal Path admits an FPTRAS when parameterized by the maximum length of a temporal (s, z)-path.

Problems:

- Not straightforward to approximate number of temporal (s, z)-path that visit a vertex v.

Approximation Results

Theorem

There is no FPRAS for **#Temporal Path** unless NP$=$BPP.

Theorem

#Temporal Path admits an FPTRAS when parameterized by the maximum length of a temporal (s, z)-path.

Problems:

- Not straightforward to approximate number of temporal (s, z)-path that visit a vertex v.
- Not straightforward to approximate temporal betweenness.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.
- Presumably not in FPT for the feedback vertex number of the underlying graph.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.
- Presumably not in FPT for the feedback vertex number of the underlying graph.
- FPT algorithms for several “distance to forest”-parameterizations.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.
- Presumably not in FPT for the feedback vertex number of the underlying graph.
- FPT algorithms for several “distance to forest”-parameterizations.

Future Work:

- Investigate vertex cover number of the underlying graph as a parameter.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.
- Presumably not in FPT for the feedback vertex number of the underlying graph.
- FPT algorithms for several “distance to forest”-parameterizations.

Future Work:

- Investigate vertex cover number of the underlying graph as a parameter.
- Most results seem also to work for restless temporal paths or delay-robust routes.

Conclusion and Future Research

Summary:

- To the best of our knowledge we initiated exploring the parameterized and approximation complexity landscape of temporal path counting.
- Solvable in polynomial time if underlying graph is a forest.
- Presumably not in FPT for the feedback vertex number of the underlying graph.
- FPT algorithms for several “distance to forest”-parameterizations.

Future Work:

- Investigate vertex cover number of the underlying graph as a parameter.
- Most results seem also to work for restless temporal paths or delay-robust routes.

Link to arXiv.

Thank you!