Realizing Temporal Transportation Trees

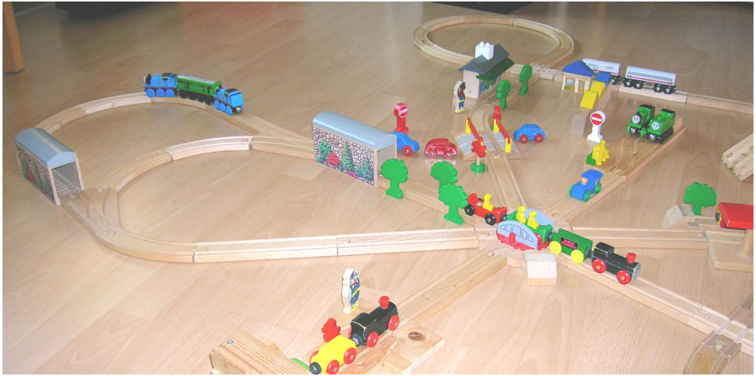
George B. Mertzios   Hendrik Molter   Paul G. Spirakis

Department of Computer Science, Ben-Gurion University of the Negev, Israel

Algorithmic Aspects of Temporal Graphs VII

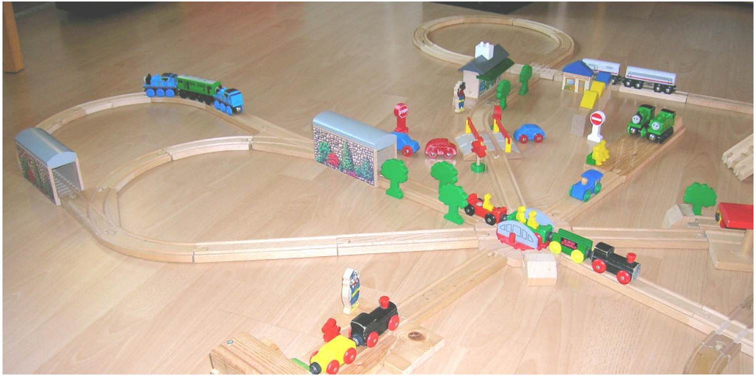
Problem Setting

Imagine you are given a train track system.



Problem Setting

Imagine you are given a train track system.

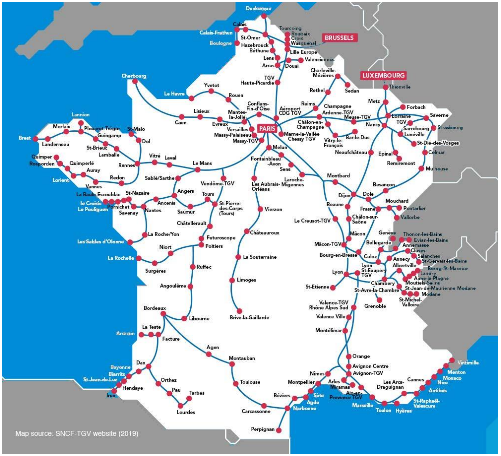


Your task is to design a (periodic) schedule for the trains, such that travel times are sufficiently fast.

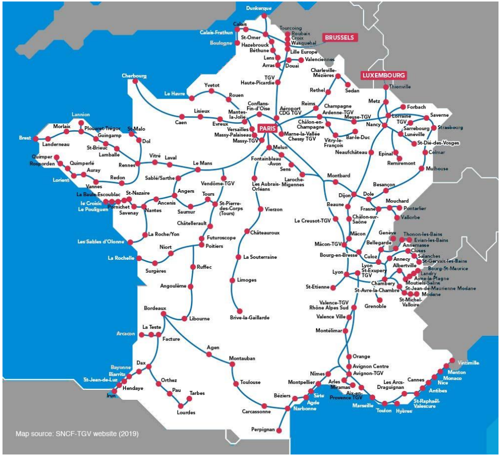
Restriction: Transportation Trees

Restriction: Train track system has a tree structure.

Restriction: Train track system has a tree structure.

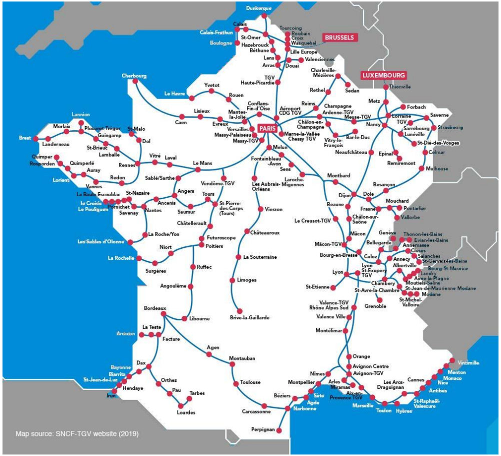


Restriction: Train track system has a tree structure.



Restriction: Transportation Trees

Restriction: Train track system has a tree structure.



Problem Definition

Temporal Tree Realization (TTR)

Problem Definition

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.

Problem Definition

<u>Temporal Tree Realization (TTR)</u>

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.

Problem Definition

<u>Temporal Tree Realization (TTR)</u>

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Problem Definition

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Simplifying Assumptions:

- Each connection (edge) is scheduled **once per period**.

Problem Definition

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Simplifying Assumptions:

- Each connection (edge) is scheduled **once per period**.
- Connections are **bidirectional** (undirected) and have **traversal time one** (strict setting).

Problem Definition

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Simplifying Assumptions:

- Each connection (edge) is scheduled **once per period**.
- Connections are **bidirectional** (undirected) and have **traversal time one** (strict setting).

Task:

Find a label $t_e \in \{1, \dots, \Delta\}$ for each edge $e \in E$ such that for all $u, v \in V$:

- Travel time from u to v is **at most** $D(u, v)$.

Problem Definition

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Simplifying Assumptions:

- Each connection (edge) is scheduled **once per period**.
- Connections are **bidirectional** (undirected) and have **traversal time one** (strict setting).

Task:

Find a label $t_e \in \{1, \ldots, \Delta\}$ for each edge $e \in E$ such that for all $u, v \in V$:

- Travel time from u to v is **at most** $D(u, v)$.

Travel time:

Labels define **periodic temporal graph** $\mathcal{G} = (V, E, \lambda)$ where for all $e \in E$:

$$\lambda(e) = \{t_e, t_e + \Delta, t_e + 2\Delta, t_e + 3\Delta, \ldots\}$$

Temporal Tree Realization (TTR)

Input:

- A tree $G = (V, E)$ with n vertices.
- A $n \times n$ distance matrix D.
- A period length Δ.

Simplifying Assumptions:

- Each connection (edge) is scheduled **once per period**.
- Connections are **bidirectional** (undirected) and have **traversal time one** (strict setting).

Task:

Find a label $t_e \in \{1, \ldots, \Delta\}$ for each edge $e \in E$ such that for all $u, v \in V$:

- Travel time from u to v is **at most** $D(u, v)$.

Travel time:

Labels define **periodic temporal graph** $\mathcal{G} = (V, E, \lambda)$ where for all $e \in E$:

$$\lambda(e) = \{t_e, t_e + \Delta, t_e + 2\Delta, t_e + 3\Delta, \ldots\}$$

Travel time $\delta(u, v)$ from u to v is **duration** of **fastest** temporal path from u to v in $\mathcal{G}$.

Example

Duration of a fastest path: Last label $-$ first label $+$ one.

Example

Duration of a fastest path: Last label $-$ first label $+$ one.

$\Delta = 4$

Example

Duration of a fastest path: Last label − first label + one.

$\Delta = 4$

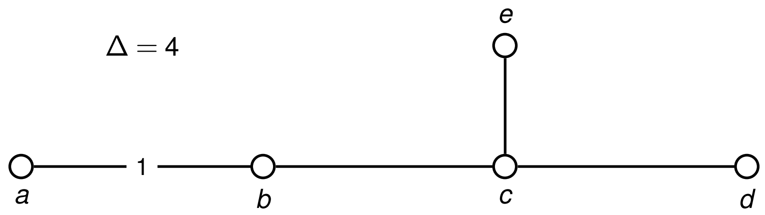
Distance matrix (excerpt):

- $D(a, c) = 3$
- $D(a, d) = 6$
- $D(d, a) = 5$
- $D(b, e) = 3$
- $D(e, b) = 3$

Actual travel times:

Example

Duration of a fastest path: Last label $-$ first label $+$ one.



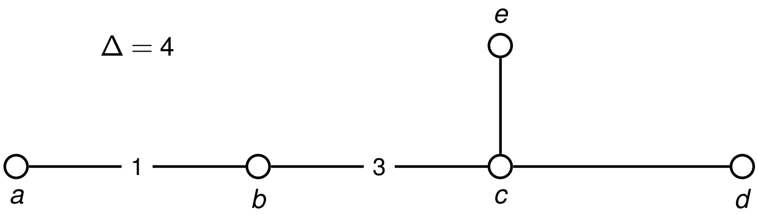
Distance matrix (excerpt):

- $D(a, c) = 3$
- $D(a, d) = 6$
- $D(d, a) = 5$
- $D(b, e) = 3$
- $D(e, b) = 3$

Actual travel times:

Example

Duration of a fastest path: Last label $-$ first label $+$ one.



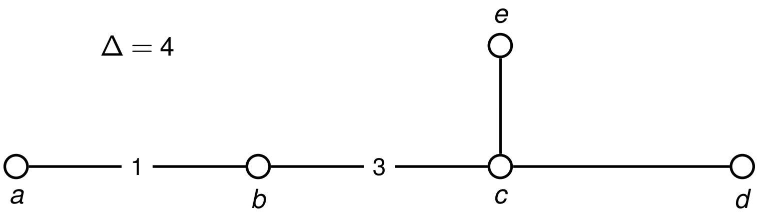
Distance matrix (excerpt):

- $D(a, c) = 3$
- $D(a, d) = 6$
- $D(d, a) = 5$
- $D(b, e) = 3$
- $D(e, b) = 3$

Actual travel times:

Example

Duration of a fastest path: Last label − first label + one.



Distance matrix (excerpt):

- $D(a,c) = 3$
- $D(a,d) = 6$
- $D(d,a) = 5$
- $D(b,e) = 3$
- $D(e,b) = 3$

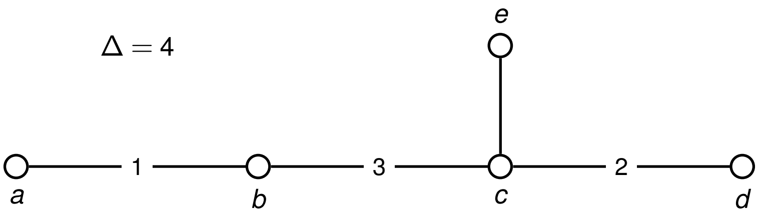
Actual travel times:

- $\delta(a,c) = 3$

Example

Duration of a fastest path: Last label $-$ first label $+$ one.

$\Delta = 4$



Distance matrix (excerpt):

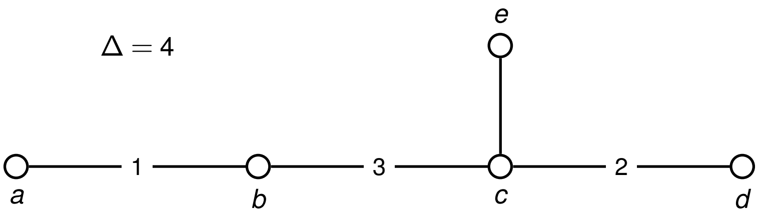
- $D(a,c) = 3$
- $D(a,d) = 6$
- $D(d,a) = 5$
- $D(b,e) = 3$
- $D(e,b) = 3$

Actual travel times:

- $\delta(a,c) = 3$

Example

Duration of a fastest path: Last label − first label + one.



Distance matrix (excerpt):

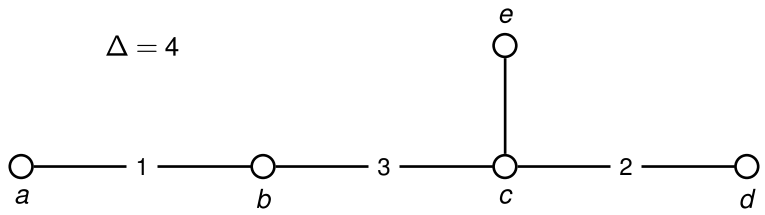
- $D(a,c) = 3$
- $D(a,d) = 6$
- $D(d,a) = 5$
- $D(b,e) = 3$
- $D(e,b) = 3$

Actual travel times:

- $\delta(a,c) = 3$
- $\delta(a,d) = 6$

Example

Duration of a fastest path: Last label − first label + one.



Distance matrix (excerpt):

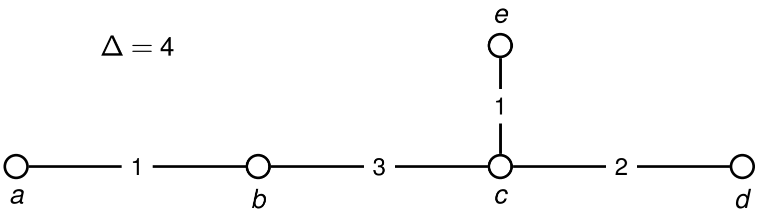
- $D(a,c) = 3$
- $D(a,d) = 6$
- $D(d,a) = 5$
- $D(b,e) = 3$
- $D(e,b) = 3$

Actual travel times:

- $\delta(a,c) = 3$
- $\delta(a,d) = 6$
- $\delta(d,a) = 4$

Example

Duration of a fastest path: Last label − first label + one.



Distance matrix (excerpt):

- $D(a, c) = 3$
- $D(a, d) = 6$
- $D(d, a) = 5$
- $D(b, e) = 3$
- $D(e, b) = 3$

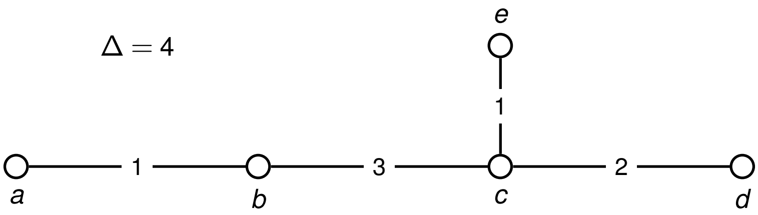
Actual travel times:

- $\delta(a, c) = 3$
- $\delta(a, d) = 6$
- $\delta(d, a) = 4$

Example

Duration of a fastest path: Last label $-$ first label $+$ one.

$\Delta = 4$



Distance matrix (excerpt):

- $D(a,c) = 3$
- $D(a,d) = 6$
- $D(d,a) = 5$
- $D(b,e) = 3$
- $D(e,b) = 3$

Actual travel times:

- $\delta(a,c) = 3$
- $\delta(a,d) = 6$
- $\delta(d,a) = 4$
- $\delta(b,e) = 3$
- $\delta(e,b) = 3$

Our Results

Theorem

TTR is **NP-hard** even on **trees with constant diameter** and if Δ is constant.

Our Results

Theorem

TTR is **NP-hard** even on **trees with constant diameter** and if Δ is constant.

Theorem

TTR is **NP-hard** even on **trees with constant maximum degree** and if Δ is constant.

Our Results

Theorem

TTR is **NP-hard** even on **trees with constant diameter** and if Δ is constant.

Theorem

TTR is **NP-hard** even on **trees with constant maximum degree** and if Δ is constant.

Theorem

TTR is **fixed-parameter tractable** w.r.t. the **number of leaves of the tree.**

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Temporal Graph Realization with Exact Distances and Multiple Labels per Period
[T. Erlebach, N. Morawietz, P. Wolf; SAND 2024]

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Temporal Graph Realization with Exact Distances and Multiple Labels per Period
[T. Erlebach, N. Morawietz, P. Wolf; SAND 2024]

- NP-hard on stars even if the number of labels per edge is a constant.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Temporal Graph Realization with Exact Distances and Multiple Labels per Period
[T. Erlebach, N. Morawietz, P. Wolf; SAND 2024]

- NP-hard on stars even if the number of labels per edge is a constant.
- FPT w.r.t. the number of vertices of the graph.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Temporal Graph Realization with Exact Distances and Multiple Labels per Period
[T. Erlebach, N. Morawietz, P. Wolf; SAND 2024]

- NP-hard on stars even if the number of labels per edge is a constant.
- FPT w.r.t. the number of vertices of the graph.
- FPT w.r.t. the vertex cover number of the graph combined with Δ.

Previous Work

Temporal Graph Realization with Exact Distances
[N. Klobas, G. B. Mertzios, H. Molter, P. G. Spirakis; SAND 2024]

- NP-hard even if Δ is constant.
- W[1]-hard w.r.t. the feedback vertex number of the graph.
- FPT w.r.t. the feedback edge number of the graph.

Temporal Graph Realization with Exact Distances and Multiple Labels per Period
[T. Erlebach, N. Morawietz, P. Wolf; SAND 2024]

- NP-hard on stars even if the number of labels per edge is a constant.
- FPT w.r.t. the number of vertices of the graph.
- FPT w.r.t. the vertex cover number of the graph combined with Δ.
- Further results.

Main Techniques for FPT Algorithm

Idea: Reduction to Mixed Integer Linear Programming.

Main Techniques for FPT Algorithm

Idea: Reduction to Mixed Integer Linear Programming.

Mixed Integer Linear Programming (MILP)

Input: A vector x of n variables of which some are considered integer variables, a constraint matrix $A \in \mathbb{R}^{m \times n}$, two vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and a target value $t \in \mathbb{R}$.

Question: Is there an assignment to the variables such that all integer variables are set to integer values, $c^\intercal x \geq t$, $Ax \leq b$, and $x \geq 0$?

Main Techniques for FPT Algorithm

Idea: Reduction to Mixed Integer Linear Programming.

Mixed Integer Linear Programming (MILP)

Input: A vector x of n variables of which some are considered integer variables, a constraint matrix $A \in \mathbb{R}^{m \times n}$, two vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and a target value $t \in \mathbb{R}$.

Question: Is there an assignment to the variables such that all integer variables are set to integer values, $c^\intercal x \geq t$, $Ax \leq b$, and $x \geq 0$?

Theorem [Lenstra '83]

MILP is fixed-parameter tractable w.r.t. the number of integer variables.

Main Techniques for FPT Algorithm

Idea: Reduction to Mixed Integer Linear Programming.

Mixed Integer Linear Programming (MILP)

Input: A vector x of n variables of which some are considered integer variables, a constraint matrix $A \in \mathbb{R}^{m \times n}$, two vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and a target value $t \in \mathbb{R}$.

Question: Is there an assignment to the variables such that all integer variables are set to integer values, $c^\intercal x \geq t$, $Ax \leq b$, and $x \geq 0$?

Theorem [Lenstra '83]

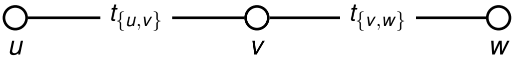
MILP is fixed-parameter tractable w.r.t. the number of integer variables.

Lemma

If the constraint matrix for the fractional variables is **totally unimodular**, then the MILP admits an optimal solution where **all variables are set to integer values**.

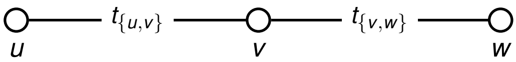
Travel Delays

Travel Delays: Let v be a vertex and u, w be neighbors of v. Let $t_{\{u,v\}}$ denote the label on edge $\{u, v\}$ and let $t_{\{v,w\}}$ denote the label on edge $\{v, w\}$.



Travel Delays

Travel Delays: Let v be a vertex and u, w be neighbors of v. Let $t_{\{u,v\}}$ denote the label on edge $\{u, v\}$ and let $t_{\{v,w\}}$ denote the label on edge $\{v, w\}$.

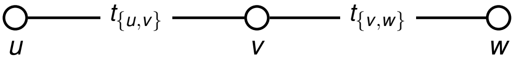


Then the **travel delay** $\tau_v^{u,w}$ from u to w at vertex v is defined as follows.

$$\tau_v^{u,w} = \begin{cases} t_{\{v,w\}} - t_{\{u,v\}}, & \text{if } t_{\{v,w\}} > t_{\{u,v\}}, \\ t_{\{v,w\}} - t_{\{u,v\}} + \Delta, & \text{otherwise.} \end{cases}$$

Travel Delays

Travel Delays: Let v be a vertex and u, w be neighbors of v. Let $t_{\{u,v\}}$ denote the label on edge $\{u, v\}$ and let $t_{\{v,w\}}$ denote the label on edge $\{v, w\}$.



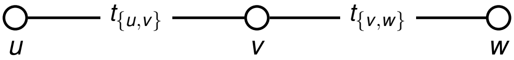
Then the **travel delay** $\tau_v^{u,w}$ from u to w at vertex v is defined as follows.

$$\tau_v^{u,w} = \begin{cases} t_{\{v,w\}} - t_{\{u,v\}}, & \text{if } t_{\{v,w\}} > t_{\{u,v\}}, \\ t_{\{v,w\}} - t_{\{u,v\}} + \Delta, & \text{otherwise.} \end{cases}$$

Duration of a fastest path: Sum of travel delays $+$ one.

Travel Delays

Travel Delays: Let v be a vertex and u, w be neighbors of v. Let $t_{\{u,v\}}$ denote the label on edge $\{u, v\}$ and let $t_{\{v,w\}}$ denote the label on edge $\{v, w\}$.



Then the **travel delay** $\tau_v^{u,w}$ from u to w at vertex v is defined as follows.

$$\tau_v^{u,w} = \begin{cases} t_{\{v,w\}} - t_{\{u,v\}}, & \text{if } t_{\{v,w\}} > t_{\{u,v\}}, \\ t_{\{v,w\}} - t_{\{u,v\}} + \Delta, & \text{otherwise.} \end{cases}$$

Duration of a fastest path: Sum of travel delays + one.

Labels computable from travel delays!

MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s, t) \leq D(s, t)$.

MILP Formulation I

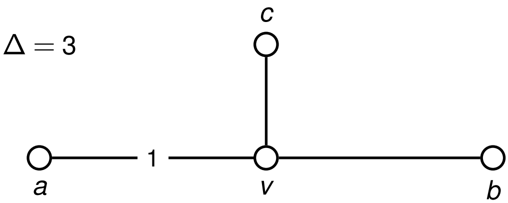
Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s, t) \leq D(s, t)$.

Problem: High degree vertices (degree larger than two).

MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

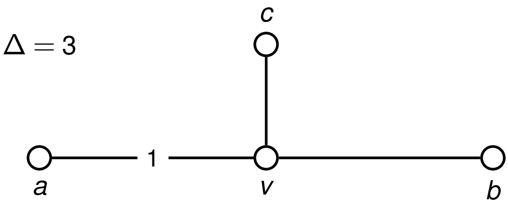


MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$

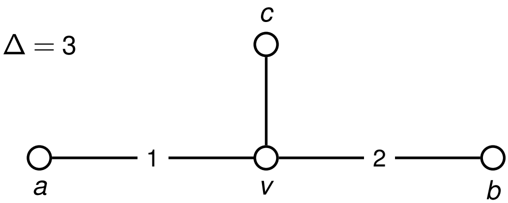


MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$



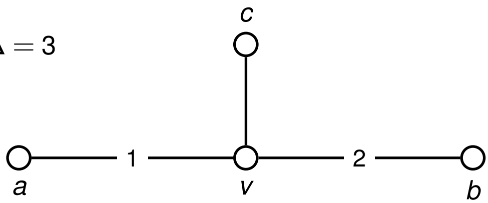
MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$
- $\tau_v^{a,c} = 1$

$\Delta = 3$

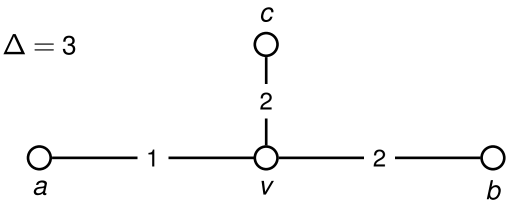


MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$
- $\tau_v^{a,c} = 1$

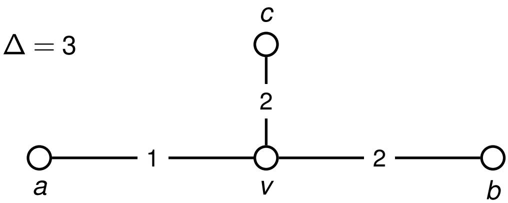


MILP Formulation I

Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$
- $\tau_v^{a,c} = 1$
- $\tau_v^{b,c} = 1$



MILP Formulation I

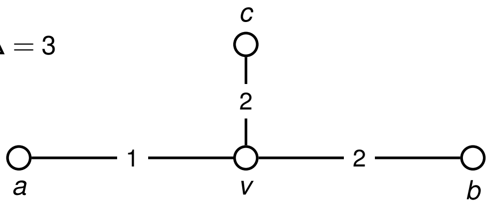
Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s, t) \leq D(s, t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$
- $\tau_v^{a,c} = 1$
- $\tau_v^{b,c} = 1$

Travel delays not realizable!

$\Delta = 3$



MILP Formulation I

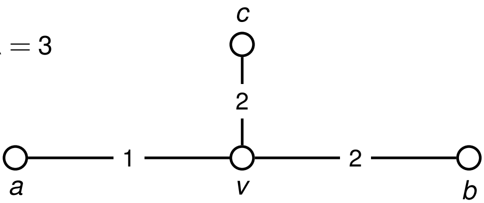
Main Idea: Create a **fractional variable** for each travel delay.
Add constraint for each vertex pair s, t checking that $\delta(s,t) \leq D(s,t)$.

Problem: High degree vertices (degree larger than two).

- $\tau_v^{a,b} = 1$
- $\tau_v^{a,c} = 1$
- $\tau_v^{b,c} = 1$

Travel delays not realizable!

$\Delta = 3$



Observation

Let G be a tree with ℓ leaves. Then the following holds.

- The maximum degree of G is at most ℓ.
- The number of vertices with degree larger than two is at most ℓ.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v.
  Assume vertices are ordered and $a < b$.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v.
  Assume vertices are ordered and $a < b$.
- Create an **integer variable** for each label of an edge incident with a high degree vertex.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v.
  Assume vertices are ordered and $a < b$.
- Create an **integer variable** for each label of an edge incident with a high degree vertex.
- For each high-degree vertex, guess ordering of labels of edges incident with the vertex.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v. Assume vertices are ordered and $a < b$.
- Create an **integer variable** for each label of an edge incident with a high degree vertex.
- For each high-degree vertex, guess ordering of labels of edges incident with the vertex.
- Use integer variables to ensure travel delays of high degree vertices are realizable.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v.
  Assume vertices are ordered and $a < b$.
- Create an **integer variable** for each label of an edge incident with a high degree vertex.
- For each high-degree vertex, guess ordering of labels of edges incident with the vertex.
- Use integer variables to ensure travel delays of high degree vertices are realizable.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

MILP Formulation II

Main Idea:

- Create a **fractional variable** $x_v^{a,b}$ for each travel delay from a to b at vertex v.
  Assume vertices are ordered and $a < b$.
- Create an **integer variable** for each label of an edge incident with a high degree vertex.
- For each high-degree vertex, guess ordering of labels of edges incident with the vertex.
- Use integer variables to ensure travel delays of high degree vertices are realizable.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) + \sum_{v \text{ has high deg and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has high deg and is traversed backward}} (\Delta - x_v^{a,b}) \leq D(s,t) - 1$$

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) \leq D(s,t) - 1 + C$$

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) \le D(s,t) - 1 + C$$

Observation: Duration constraints form a **network matrix**, which are **totally unimodular**.

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) \leq D(s,t) - 1 + C$$

Observation: Duration constraints form a **network matrix**, which are **totally unimodular**.

Informal Requirement:

- Every constraint corresponds to a path in a directed tree. Variables correspond to arcs.

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) \leq D(s,t) - 1 + C$$

Observation: Duration constraints form a **network matrix**, which are **totally unimodular**.

Informal Requirement:

- Every constraint corresponds to a path in a directed tree. Variables correspond to arcs.
- If arc appears forward, then coefficient is one.

MILP Formulation II

Observation: Variables for travel delays of high degree vertices have integer values, due to realizability constraints.

Duration Constraints: For each pair of vertices s, t we have the following constraint.

$$\sum_{v \text{ has deg 2 and is traversed forward}} x_v^{a,b} + \sum_{v \text{ has deg 2 and is traversed backward}} (\Delta - x_v^{a,b}) \leq D(s,t) - 1 + C$$

Observation: Duration constraints form a **network matrix**, which are **totally unimodular**.

Informal Requirement:

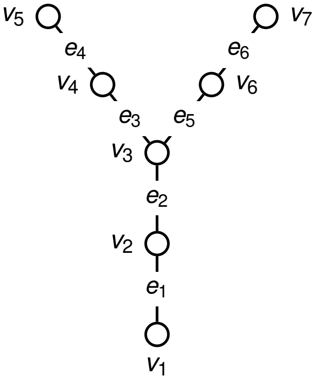
- Every constraint corresponds to a path in a directed tree. Variables correspond to arcs.
- If arc appears forward, then coefficient is one.
- If arc appears backward, then coefficient is minus one.

MILP Formulation III

Formal construction of the directed tree:

MILP Formulation III

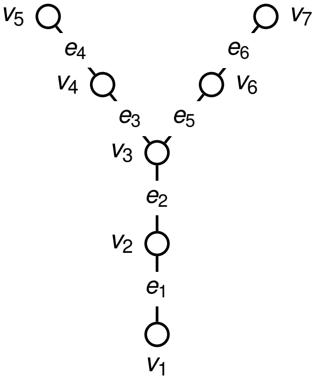
Formal construction of the directed tree:



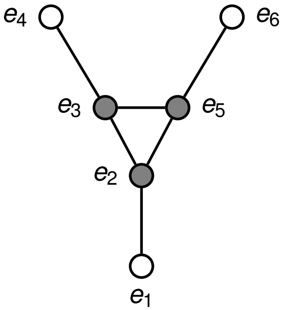
Tree G.

MILP Formulation III

Formal construction of the directed tree:



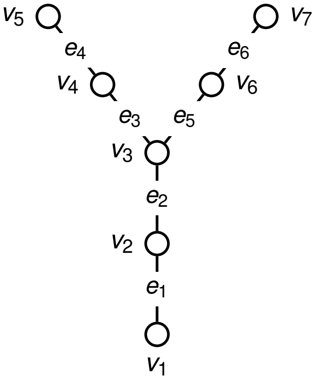
Tree G.



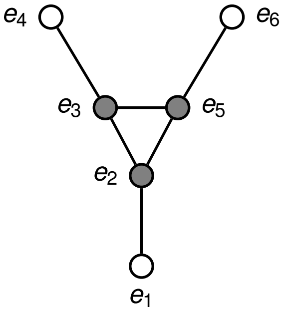
Line graph $L(G)$.

MILP Formulation III

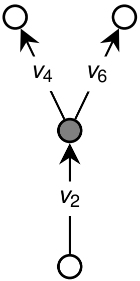
Formal construction of the directed tree:



Tree G.



Line graph $L(G)$.



Directed tree T.

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Future Work:

- How to generalize algorithmic result to multiple labels?

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Future Work:

- How to generalize algorithmic result to multiple labels?
- How to generalize algorithmic result to non-trees?

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Future Work:

- How to generalize algorithmic result to multiple labels?
- How to generalize algorithmic result to non-trees?
- Many other temporal graph realization settings are waiting to be investigated!

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Future Work:

- How to generalize algorithmic result to multiple labels?
- How to generalize algorithmic result to non-trees?
- Many other temporal graph realization settings are waiting to be investigated!

Link to arXiv.

Conclusion and Future Research

Summary:

- New temporal graph realization setting (upper bounds).
- Surprising differences in computational complexity to setting with exact distances.
- Technique for showing that MILPs admit optimal integer solutions (all variables) useful in many other contexts.

Future Work:

- How to generalize algorithmic result to multiple labels?
- How to generalize algorithmic result to non-trees?
- Many other temporal graph realization settings are waiting to be investigated!

Link to arXiv.

Thank you!