Revisiting ILP Models for Exact Crossing Minimization in Storyline Drawings

Alexander Dobler[1], Michael Jünger[2], Paul J. Jünger[3], Julian Meffert[3], Petra Mutzel[3], Martin Nöllenburg[1]

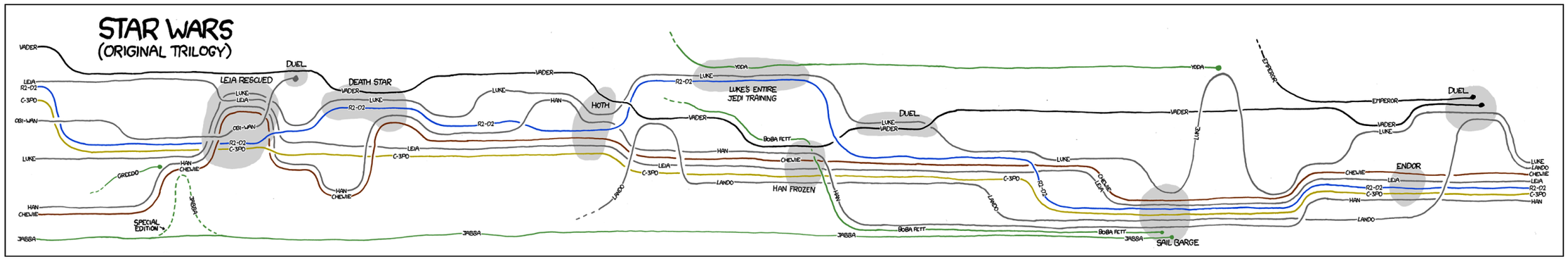
September 18, 2024 · GD 2024

[1] TU WIEN ac ALGORITHMS AND COMPLEXITY GROUP

[3] UNIVERSITÄT BONN

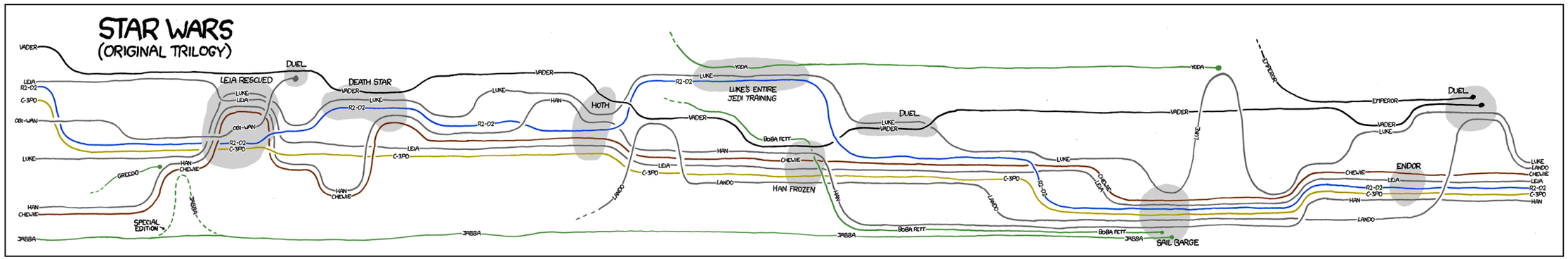
[2] UNIVERSITÄT ZU KÖLN

Storylines



©xkcd

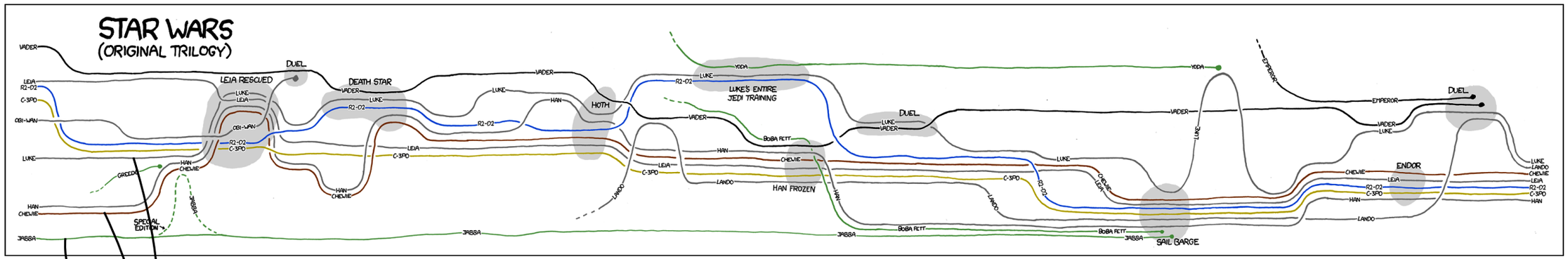
Storylines



Time

©xkcd

Storylines

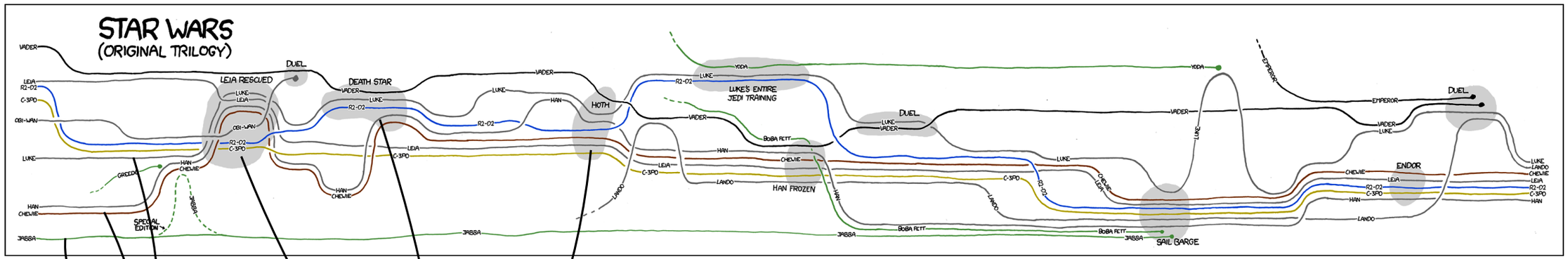


©xkcd

Characters

Time

Storylines



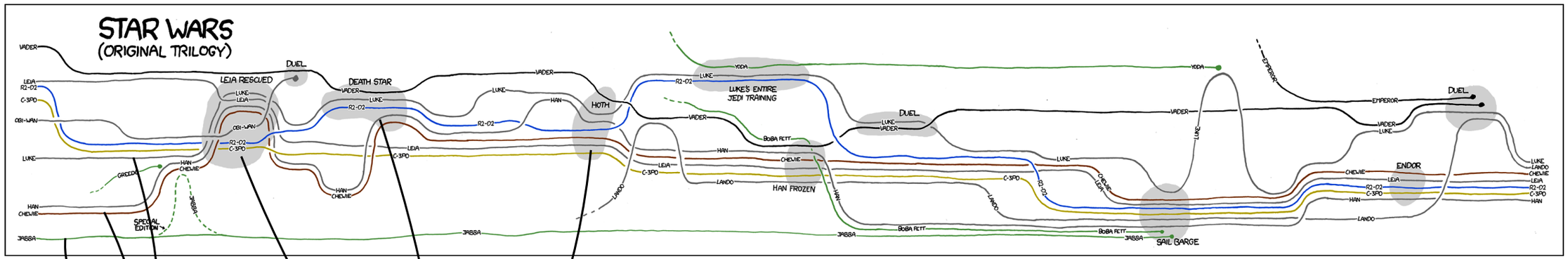
©xkcd

Characters

Time

Interactions

Storylines



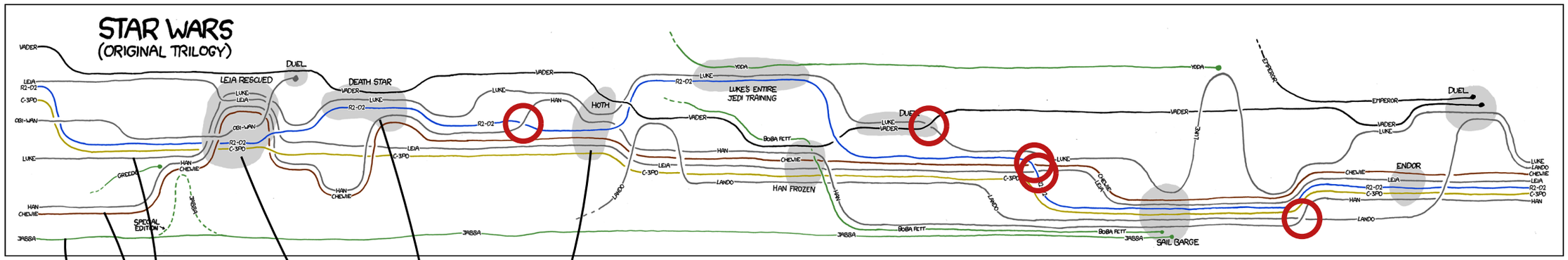
Characters

Interactions

Time

Visualization for **books**, **movies**, **publication data**, . . .

Storylines



©xkcd

Characters

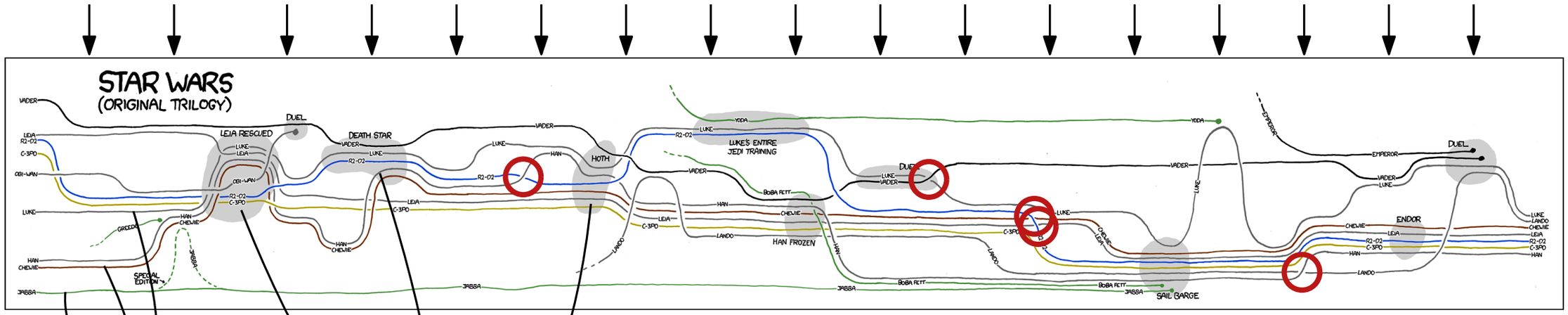
Interactions

Time

Visualization for **books**, **movies**, **publication data**, . . .

Goal: Minimize **crossings**

Storylines



©xkcd

Characters

Interactions

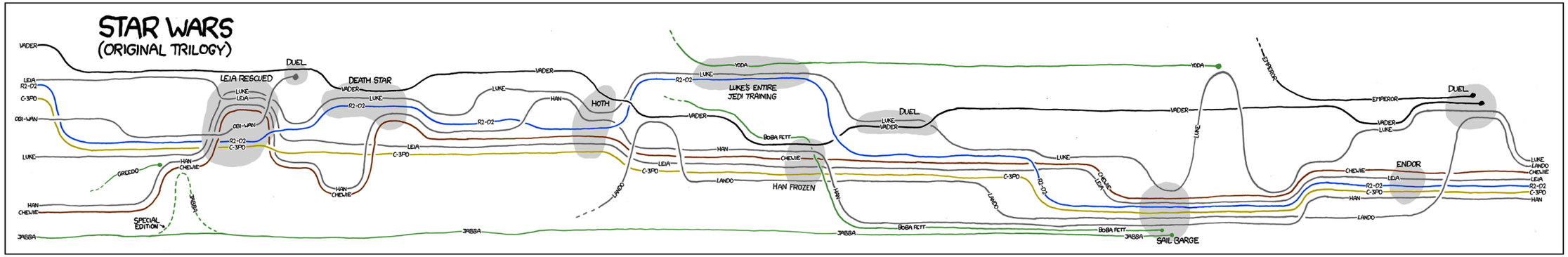
Time

Visualization for **books**, **movies**, **publication data**, . . .

Goal: Minimize **crossings**

Degree of freedom: vertical **order** of character curves over time

Previous Work on Crossing Minimization



- [Ogawa and Ma, 2010]: greedy algorithm
- [Tanahashi and Ma, 2012]: genetic algorithm
- [Liu, Wu, Wei, Liu, and Liu, 2013]: pipeline approach
- [Kostitsyna, Nöllenburg, Polishchuk, Schulz, and Strash, 2015]: NP-hardness and fixed-parameter tractability by #characters
- [Gronemann, Jünger, Liers, Mambelli, 2016]: exact integer programming formulations and experimental evaluation of one formulation

Our Contribution

[GJLM, 2016]: exact integer programming formulations and experimental evaluation of algorithm based on one formulation

Our Contribution

[GJLM, 2016]: exact integer programming formulations and experimental evaluation of algorithm based on one formulation

We enrich the **simple formulations** by

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

Our Contribution

[GJLM, 2016]: exact integer programming formulations and experimental evaluation of algorithm based on one formulation

We enrich the **simple formulations** by

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

and obtain **faster exact algorithms** which we test in an **experimental evaluation**

Our Contribution

[GJLM, 2016]: exact integer programming formulations and experimental evaluation of algorithm based on one formulation

We enrich the **simple formulations** by

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

and obtain **faster exact algorithms** which we test in an **experimental evaluation**

We also provide an extended **open source benchmark dataset**, which may contain your favourite movie

Our Contribution

[GJLM, 2016]: exact integer programming formulations and experimental evaluation of algorithm based on one formulation

We enrich the **simple formulations** by

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

and obtain **faster exact algorithms** which we test in an **experimental evaluation**

We also provide an extended **open source benchmark dataset**, which may contain your favourite movie
Barbie, Oppenheimer, Harry Potter, Titanic, ...

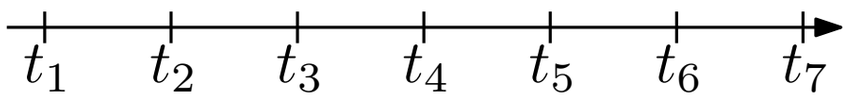
Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

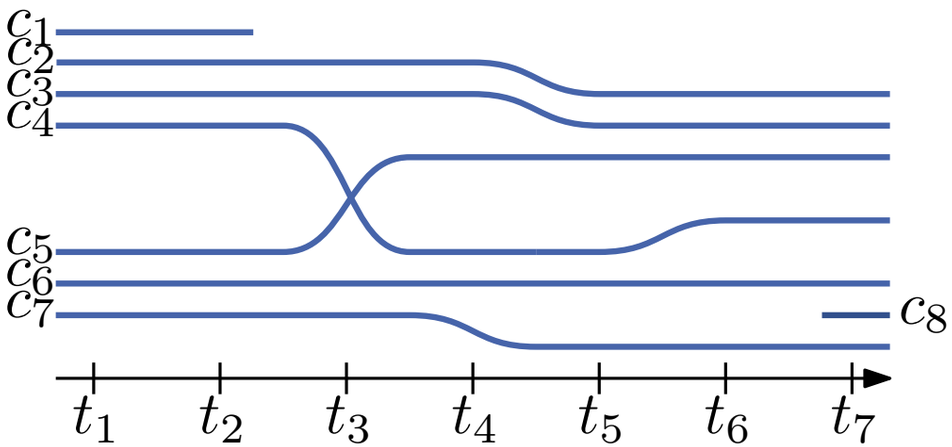
- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*



Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

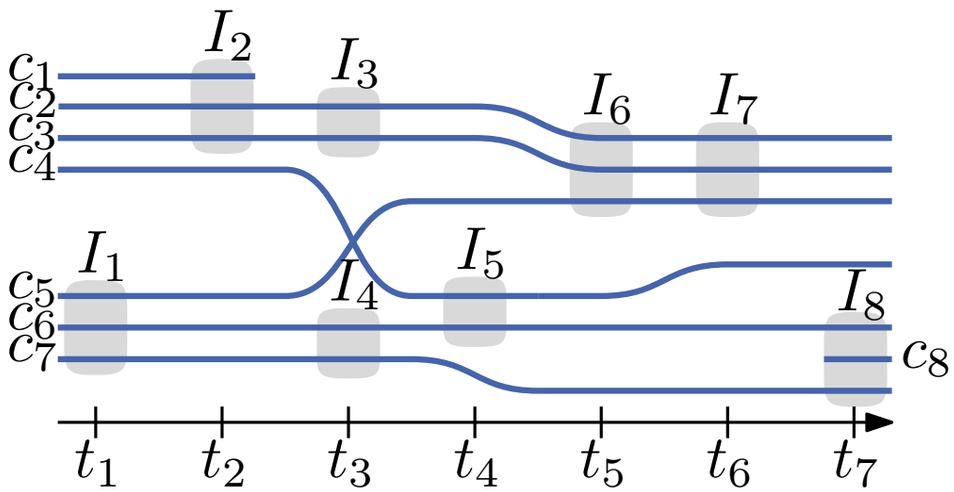
- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*



Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

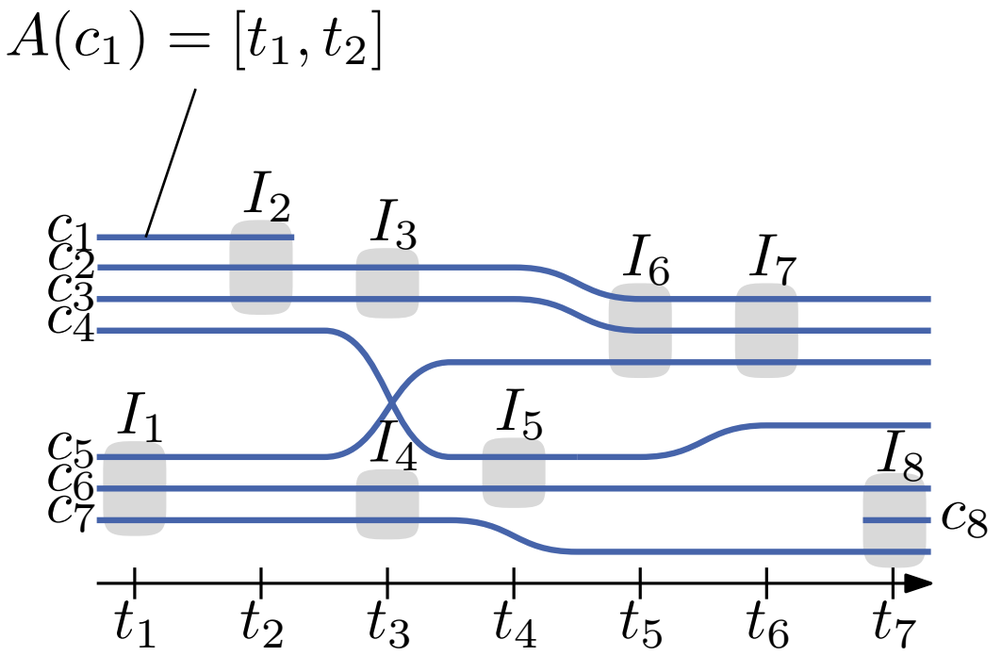
- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$



Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

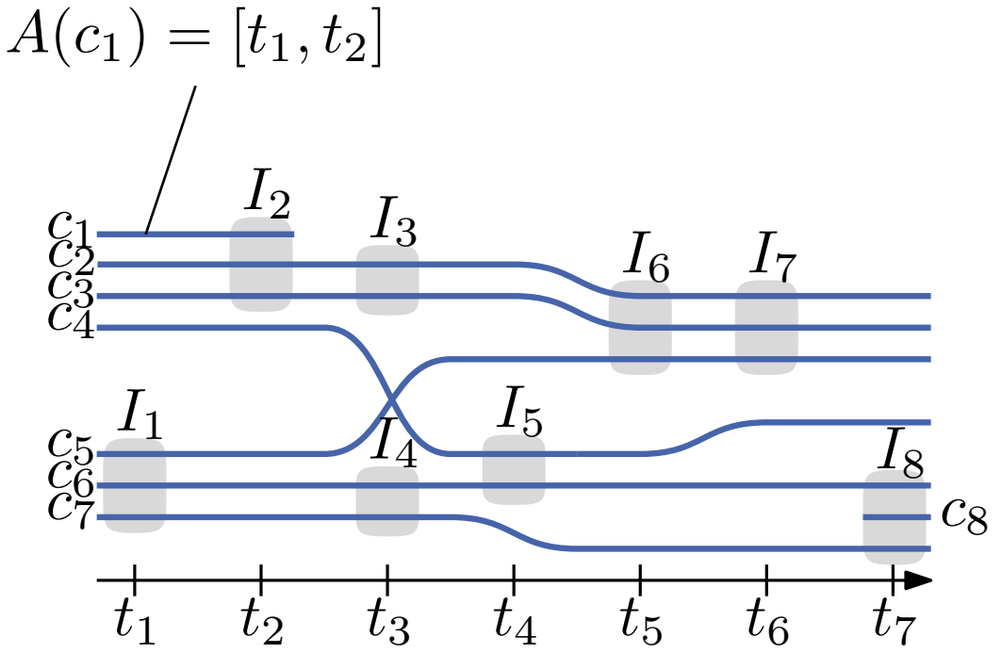
- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



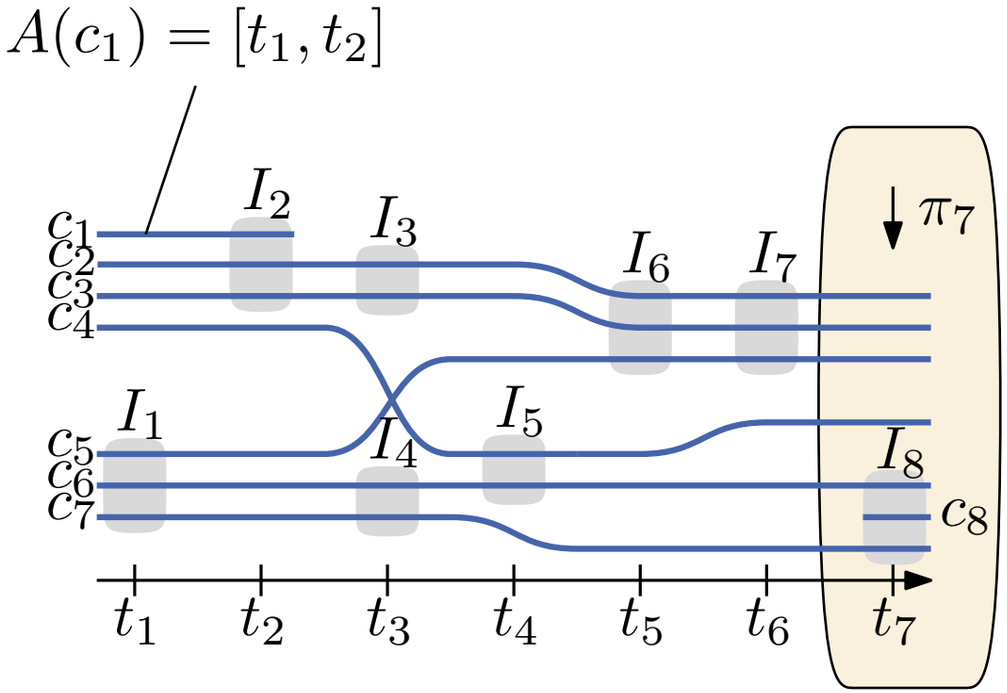
Solution:

A sequence $(\pi_1, \ldots, \pi_\ell)$ of permutations of respective active characters s.t. interaction characters are consecutive in the respective permutations

Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



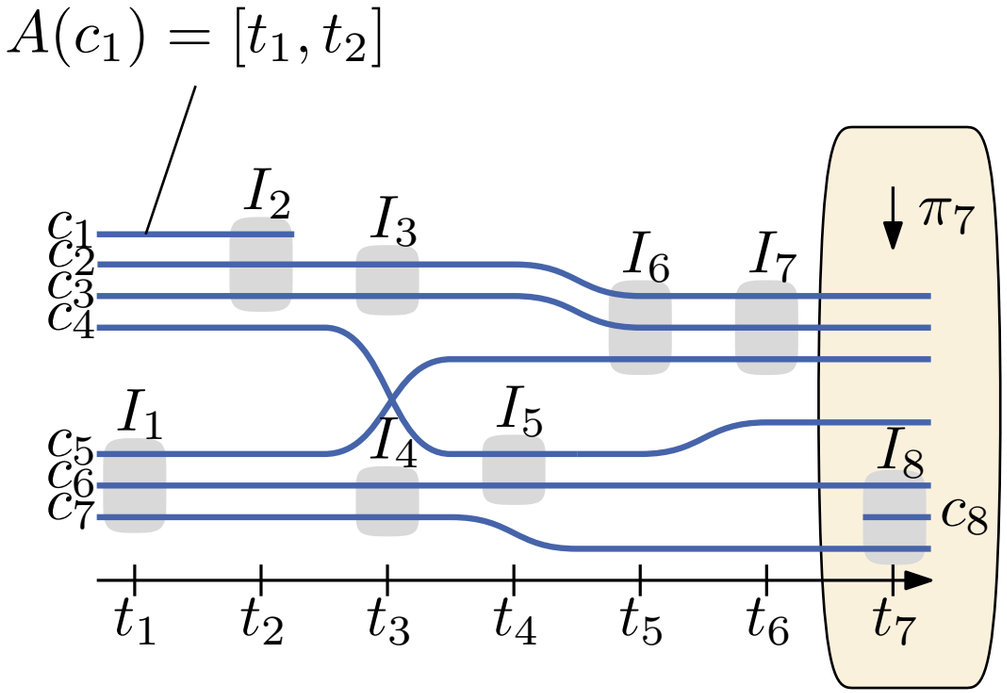
Solution:

A sequence $(\pi_1, \ldots, \pi_\ell)$ of permutations of respective active characters s.t. interaction characters are consecutive in the respective permutations

Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



Solution:

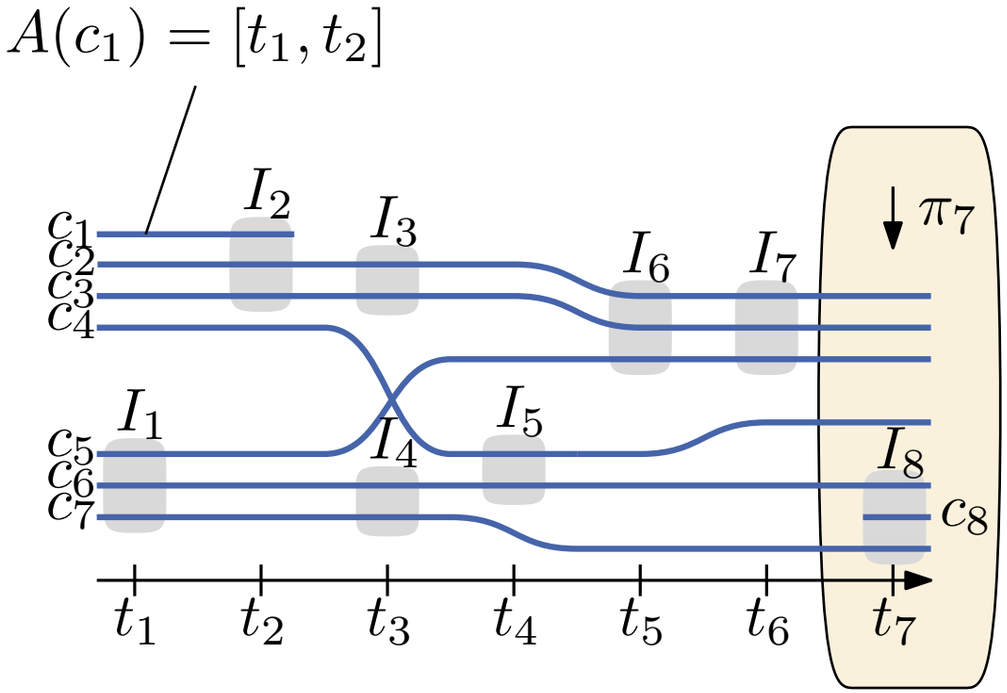
A sequence $(\pi_1, \ldots, \pi_\ell)$ of permutations of respective active characters s.t. interaction characters are consecutive in the respective permutations

Objective: minimize crossings

Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



Solution:

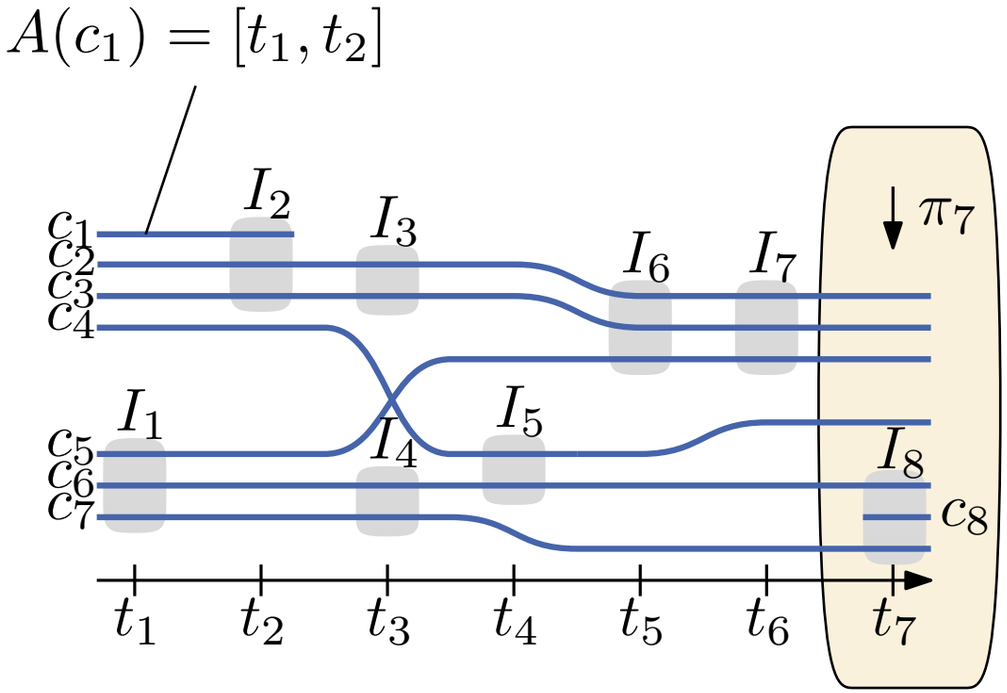
A sequence $(\pi_1, \ldots, \pi_\ell)$ of permutations of respective active characters s.t. interaction characters are consecutive in the respective permutations

Objective: minimize crossings

Formal Problem Definition

Input: $(T, \mathcal{C}, \mathcal{I}, A)$ where

- $T = \{t_1, \ldots, t_\ell\}$ is a set of totally ordered *timesteps*
- $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set *characters*
- $\mathcal{I} = \{I_1, \ldots, I_m\}$ is a set of *interactions* with a time $\mathrm{time}(I) \in T$ and characters $\mathrm{char}(I) \subseteq \mathcal{C}$ for $I \in \mathcal{I}$
- $A(c)$ for $c \in \mathcal{C}$ is a consecutive range of timesteps, the *active interval* of c



Solution:

A sequence $(\pi_1, \ldots, \pi_\ell)$ of permutations of respective active characters s.t. interaction characters are consecutive in the respective permutations

Objective: minimize crossings

combinatorial problem!

Three Integer Programming Formulations

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i

1. Model with **quadratic** objective function

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i

1. Model with **quadratic** objective function
2. **Linearized** model

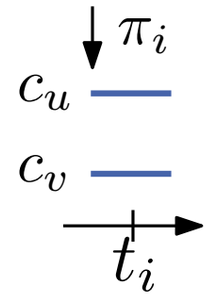
Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i

1. Model with **quadratic** objective function
2. **Linearized** model
3. **Stronger linear model** obtained by a transformation to **Max-Cut**

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i



1. Model with **quadratic** objective function
2. **Linearized** model
3. **Stronger linear model** obtained by a transformation to **Max-Cut**

[GJLM, 2016] implement this

$$\max \sum_{e \in E_M} w_e z_e \qquad \text{(CUT)}$$

$$\sum_{e \in F} z_e - \sum_{e \in C \setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M, F \subseteq C, |F| \text{ odd} \qquad \text{(CYC)}$$

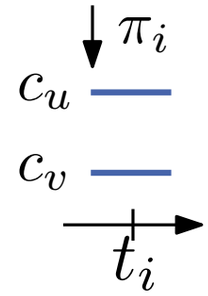
$$0 \leq z_{(v^*,c^i_{uv})} + z_{(v^*,c^i_{vw})} - z_{(v^*,c^i_{uw})} \leq 1 \quad \text{for all } i = 1,\dots,\ell; c^i_{uv}, c^i_{vw}, c^i_{uw} \in V_i \text{ with } c_u \prec_{\hat{\pi}_i} c_v \prec_{\hat{\pi}_i} c_w \qquad \text{(LOPC)}$$

$$\left.\begin{aligned} z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{vw})} \text{ if } c_u, c_v \prec_{\hat{\pi}_i} c_w \\ z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{wv})} \text{ if } c_u, c_v \succ_{\hat{\pi}_i} c_w \end{aligned}\right\} \quad \begin{aligned} &\text{for all } i = 1,\dots,\ell; I \in \mathcal{I}(t_i); \\ &c_u, c_v \in \mathrm{char}(I); c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I) \end{aligned} \qquad \text{(TRC)}$$

$$z_e \in \{0,1\} \quad \text{for all } e \in E_M \qquad \text{(BIC)}$$

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i



1. Model with **quadratic** objective function
2. **Linearized** model
3. **Stronger linear model** obtained by a transformation to **Max-Cut**

[GJLM, 2016] implement this

$$\sum_{e\in F} z_e - \sum_{e\in C\setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M,$$

$$F \subseteq C, |F| \text{ odd}$$

$$\max \sum_{e\in E_M} w_e z_e \qquad \text{(CUT)}$$

$$\sum_{e\in F} z_e - \sum_{e\in C\setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M, F \subseteq C, |F| \text{ odd} \qquad \text{(CYC)}$$

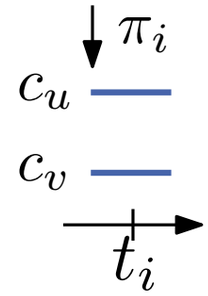
$$0 \leq z_{(v^*,c^i_{uv})} + z_{(v^*,c^i_{vw})} - z_{(v^*,c^i_{uw})} \leq 1 \quad \text{for all } i = 1,\dots,\ell; c^i_{uv}, c^i_{vw}, c^i_{uw} \in V_i \text{ with } c_u \prec_{\hat{\pi}_i} c_v \prec_{\hat{\pi}_i} c_w \qquad \text{(LOPC)}$$

$$\left.\begin{aligned} z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{vw})} \text{ if } c_u, c_v \prec_{\hat{\pi}_i} c_w \\ z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{wv})} \text{ if } c_u, c_v \succ_{\hat{\pi}_i} c_w \end{aligned}\right\} \quad \begin{aligned} &\text{for all } i = 1,\dots,\ell; I \in \mathcal{I}(t_i); \\ &c_u, c_v \in \text{char}(I); c_w \in \text{AC}(t_i) \setminus \text{char}(I) \end{aligned} \qquad \text{(TRC)}$$

$$z_e \in \{0,1\} \quad \text{for all } e \in E_M \qquad \text{(BIC)}$$

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i



1. Model with **quadratic** objective function
2. **Linearized** model
3. **Stronger linear model** obtained by a transformation to **Max-Cut**

exponential number of constraints
$\rightarrow$ branch and cut, complicated to implement

[GJLM, 2016] implement this

$$\sum_{e\in F} z_e - \sum_{e\in C\setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M,$$

$$F \subseteq C, |F| \text{ odd}$$

$$\max \sum_{e\in E_M} w_e z_e \qquad \text{(CUT)}$$

$$\sum_{e\in F} z_e - \sum_{e\in C\setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M, F \subseteq C, |F| \text{ odd} \qquad \text{(CYC)}$$

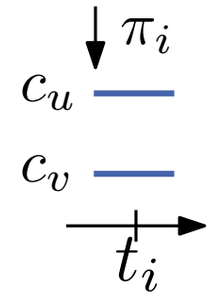
$$0 \leq z_{(v^*,c^i_{uv})} + z_{(v^*,c^i_{vw})} - z_{(v^*,c^i_{uw})} \leq 1 \quad \text{for all } i = 1,\dots,\ell; c^i_{uv}, c^i_{vw}, c^i_{uw} \in V_i \text{ with } c_u \prec_{\hat{\pi}_i} c_v \prec_{\hat{\pi}_i} c_w \qquad \text{(LOPC)}$$

$$\left.\begin{aligned} z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{vw})} \text{ if } c_u, c_v \prec_{\hat{\pi}_i} c_w \\ z_{(v^*,c^i_{uw})} = z_{(v^*,c^i_{wv})} \text{ if } c_u, c_v \succ_{\hat{\pi}_i} c_w \end{aligned}\right\} \quad \begin{aligned} &\text{for all } i = 1,\dots,\ell; I \in \mathcal{I}(t_i); \\ &c_u, c_v \in \text{char}(I); c_w \in \text{AC}(t_i) \setminus \text{char}(I) \end{aligned} \qquad \text{(TRC)}$$

$$z_e \in \{0,1\} \quad \text{for all } e \in E_M \qquad \text{(BIC)}$$

Three Integer Programming Formulations

In common: Binary variable $x_{i,u,v}$ which is 1 iff character c_u is above c_v at t_i



1. Model with **quadratic** objective function
2. **Linearized** model
3. **Stronger linear model** obtained by a transformation to **Max-Cut**

exponential number of constraints
$\rightarrow$ branch and cut, complicated to implement

[GJLM, 2016] implement this

$$\sum_{e \in F} z_e - \sum_{e \in C \setminus F} z_e \leq |F| - 1 \quad \text{for all } C \subseteq E, F \subseteq C, |F| \text{ odd}$$

We work with **simple** linear and quadratic model!

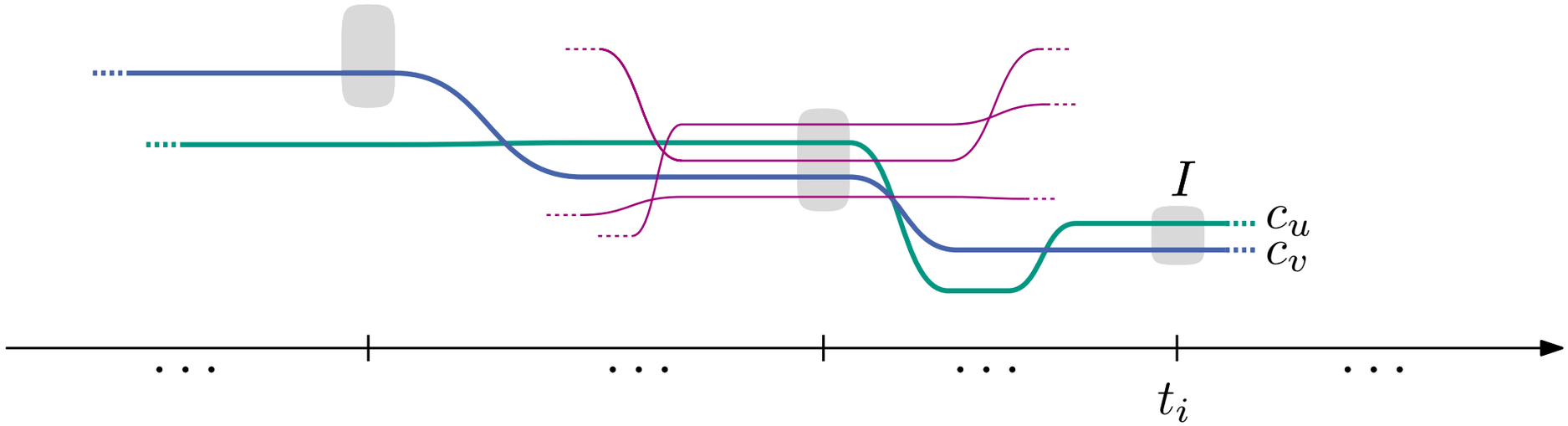
$$F \subseteq C, |F| \text{ odd}$$

Our contribution

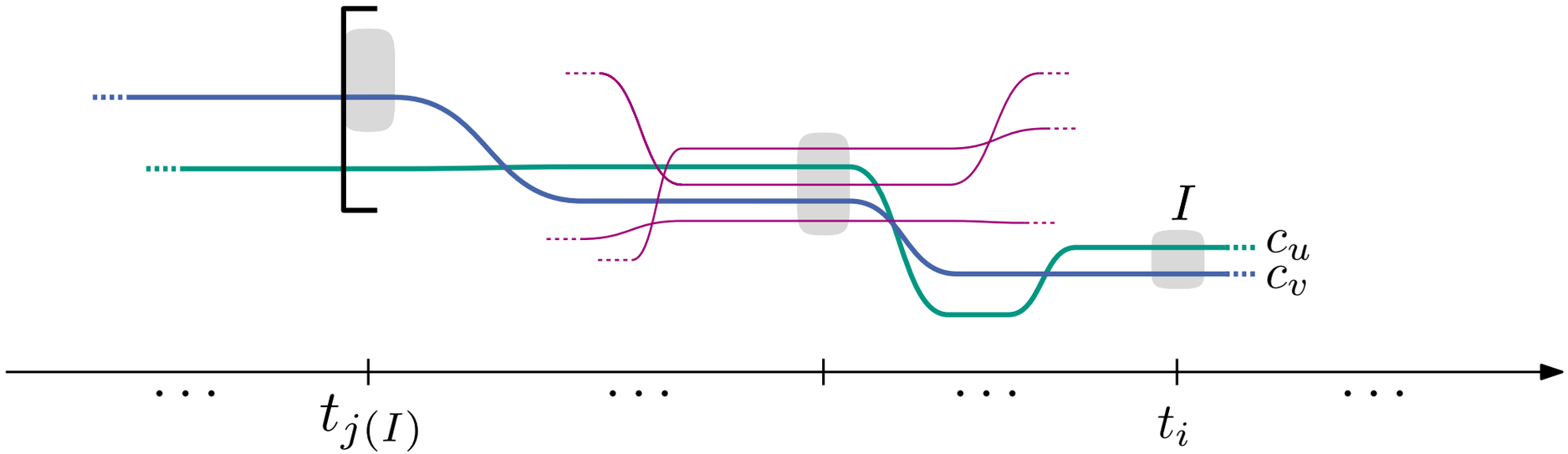
We enrich the **simple formulations** by

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

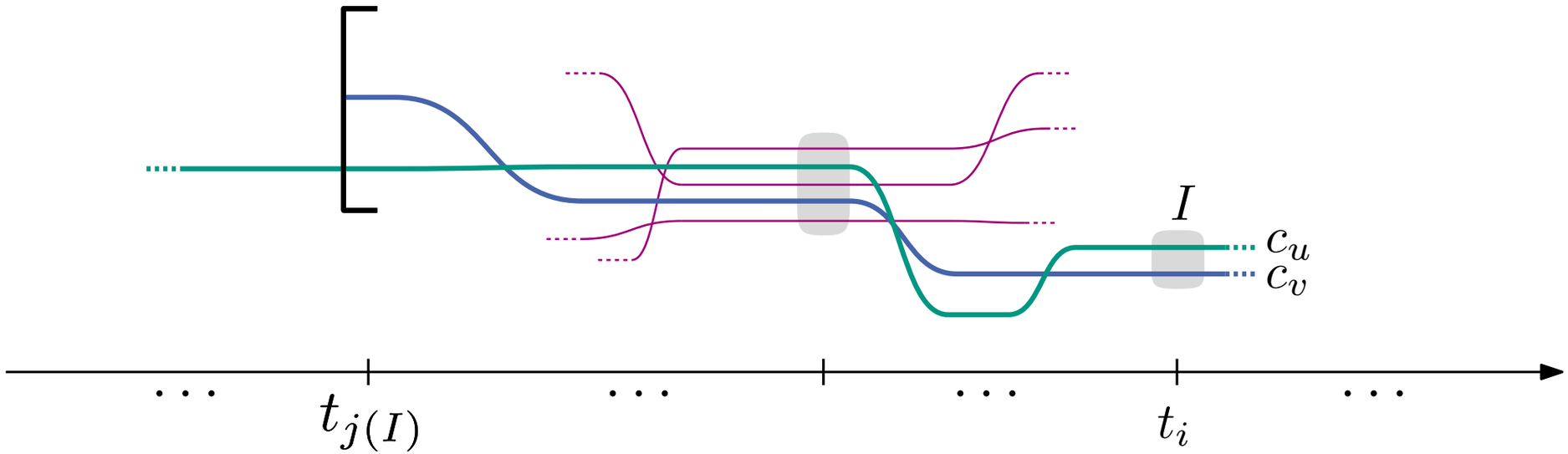
Structural Property 1



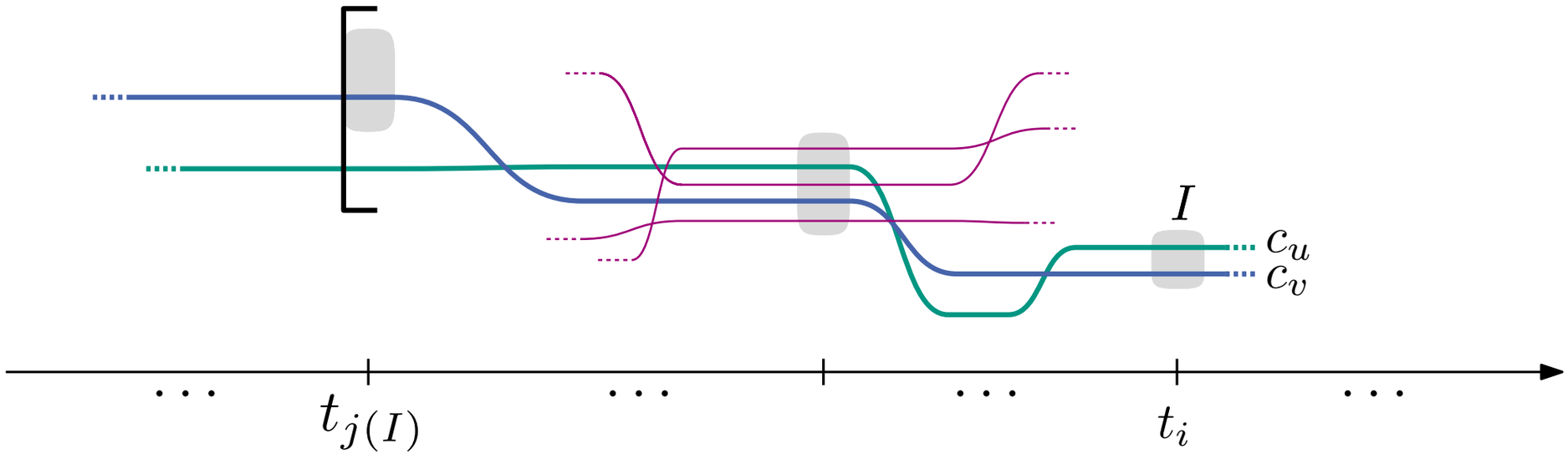
Structural Property 1



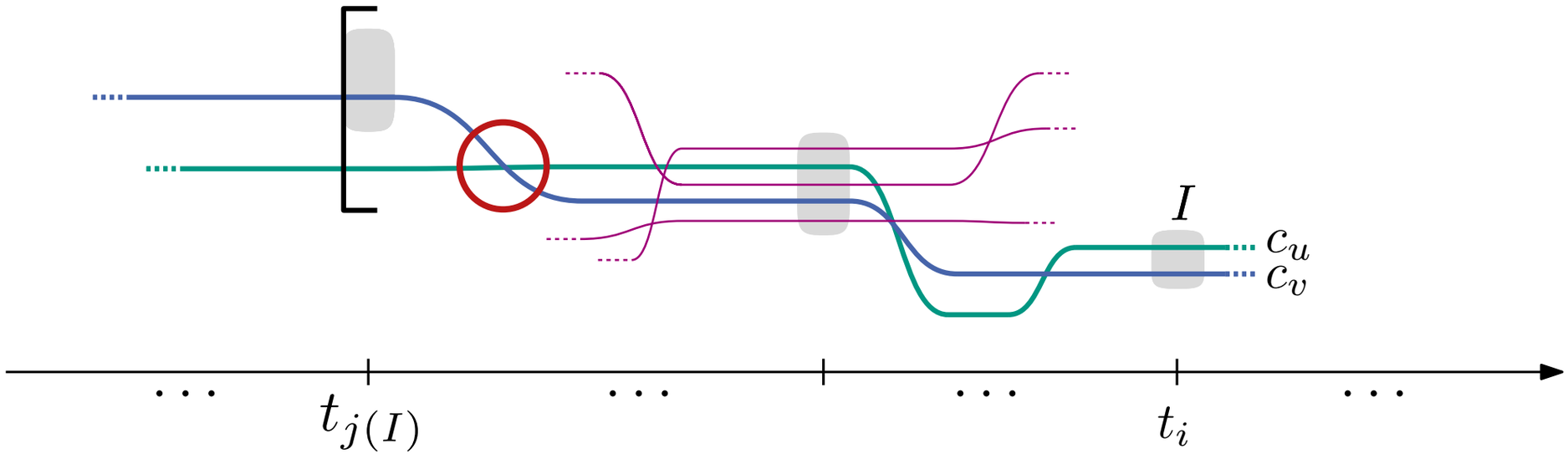
Structural Property 1



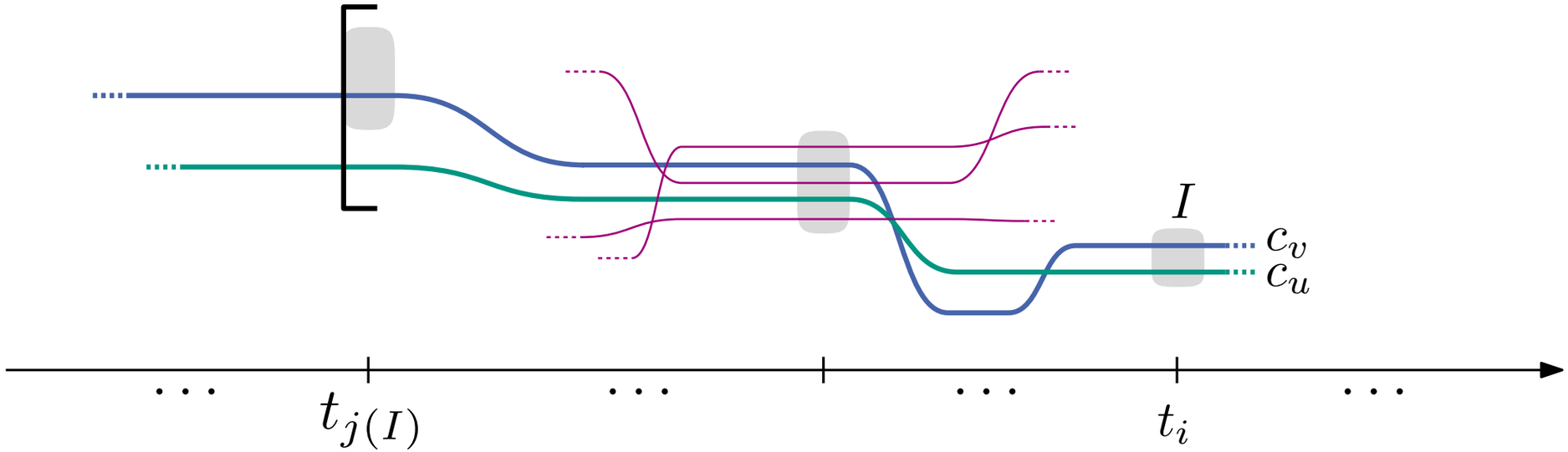
Structural Property 1



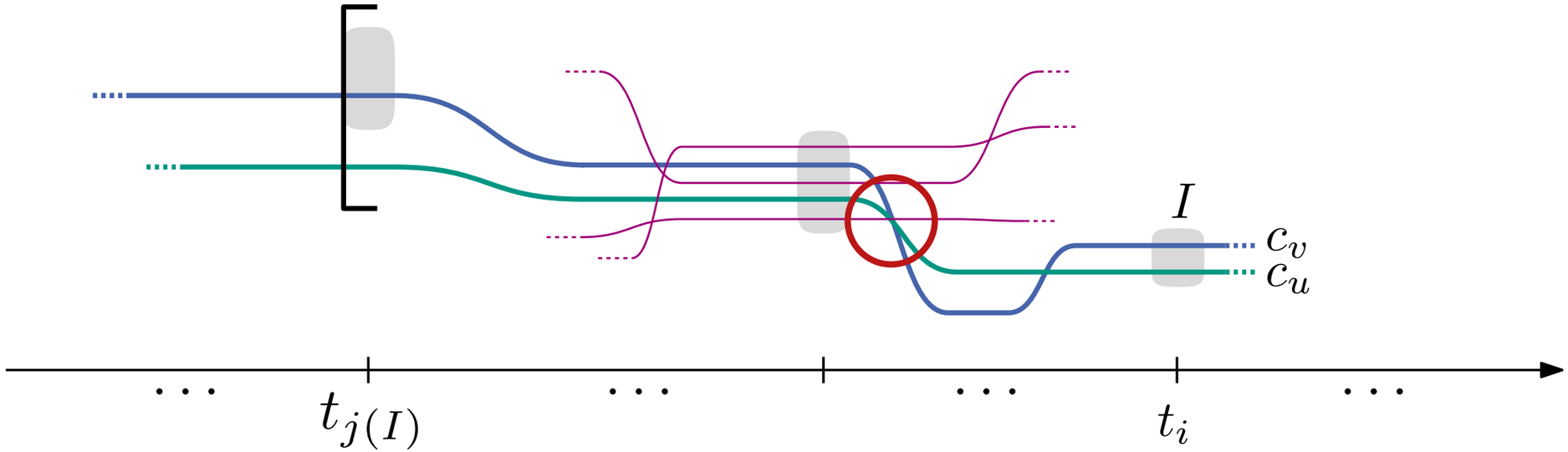
Structural Property 1



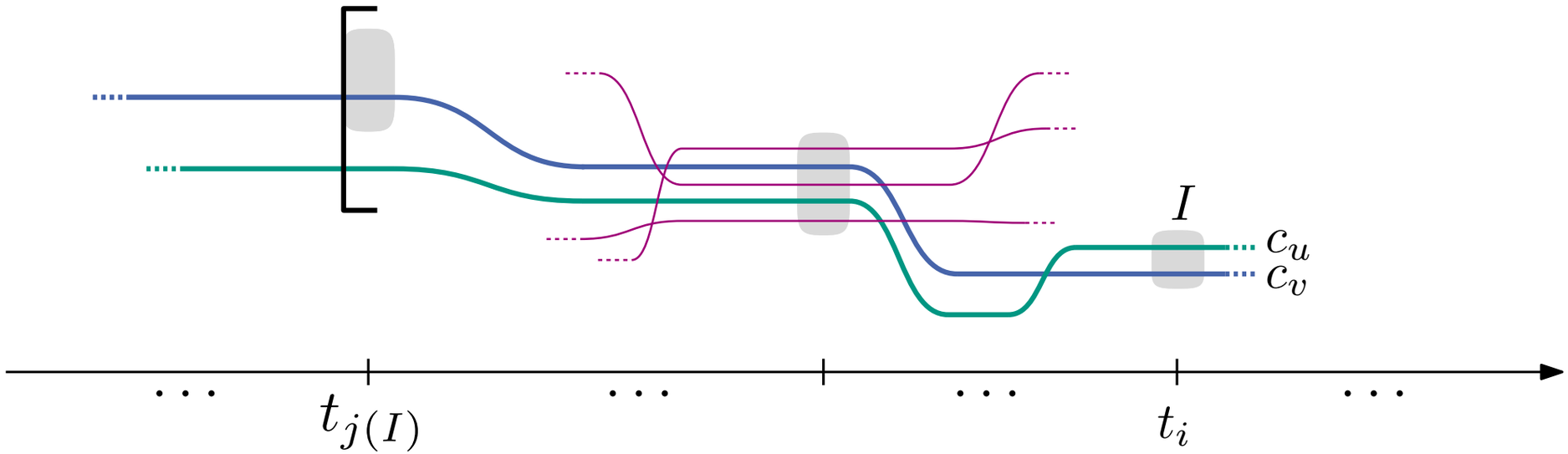
Structural Property 1



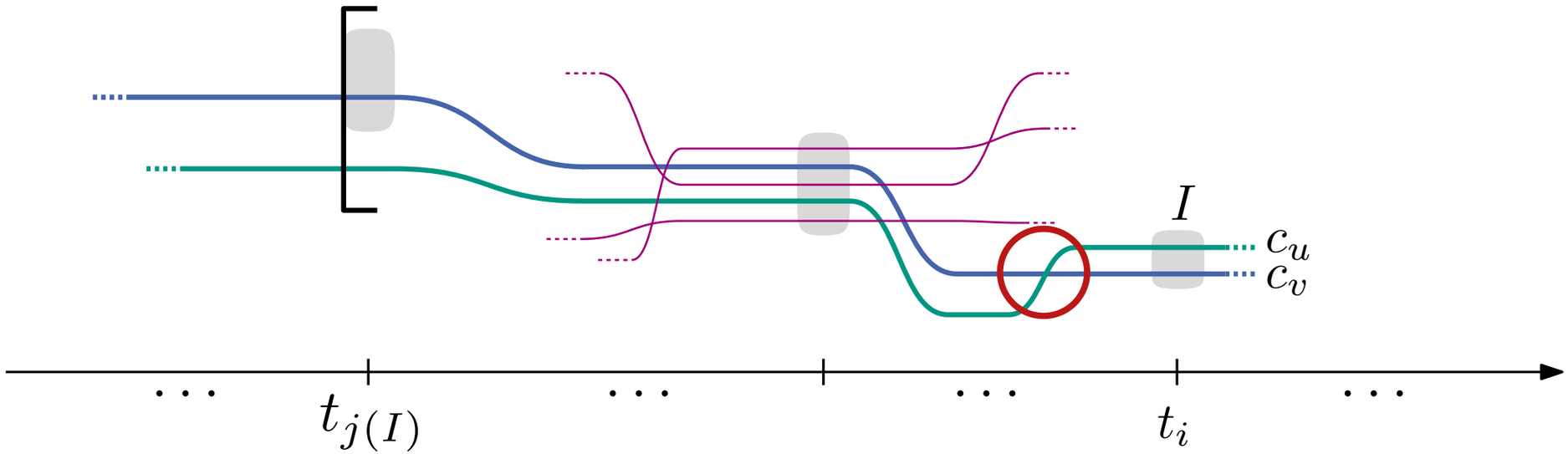
Structural Property 1



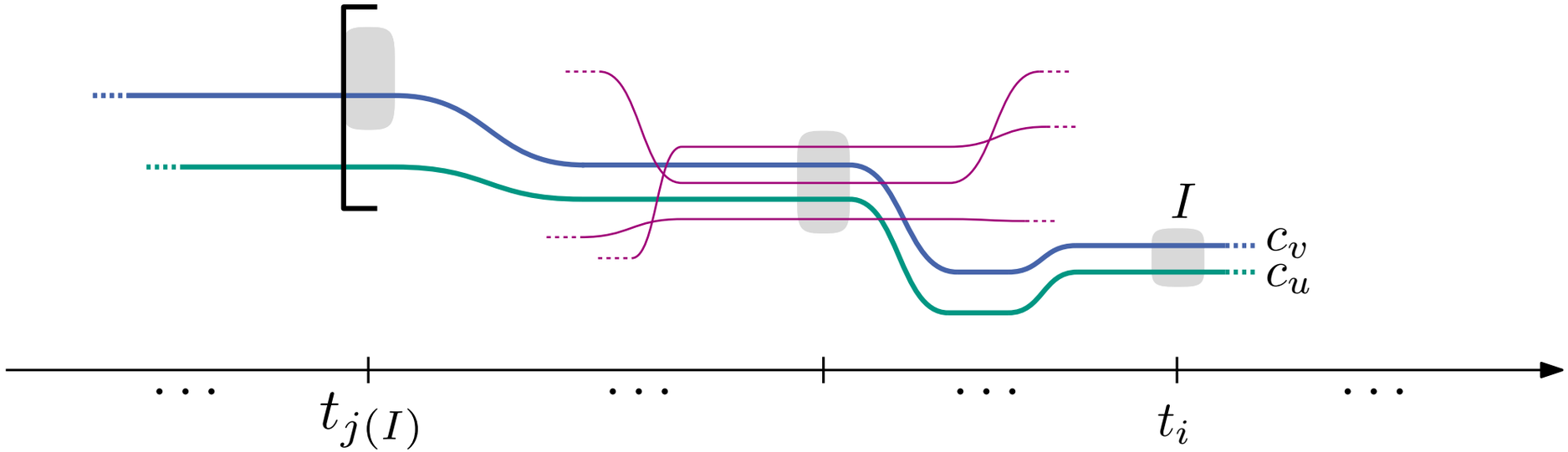
Structural Property 1



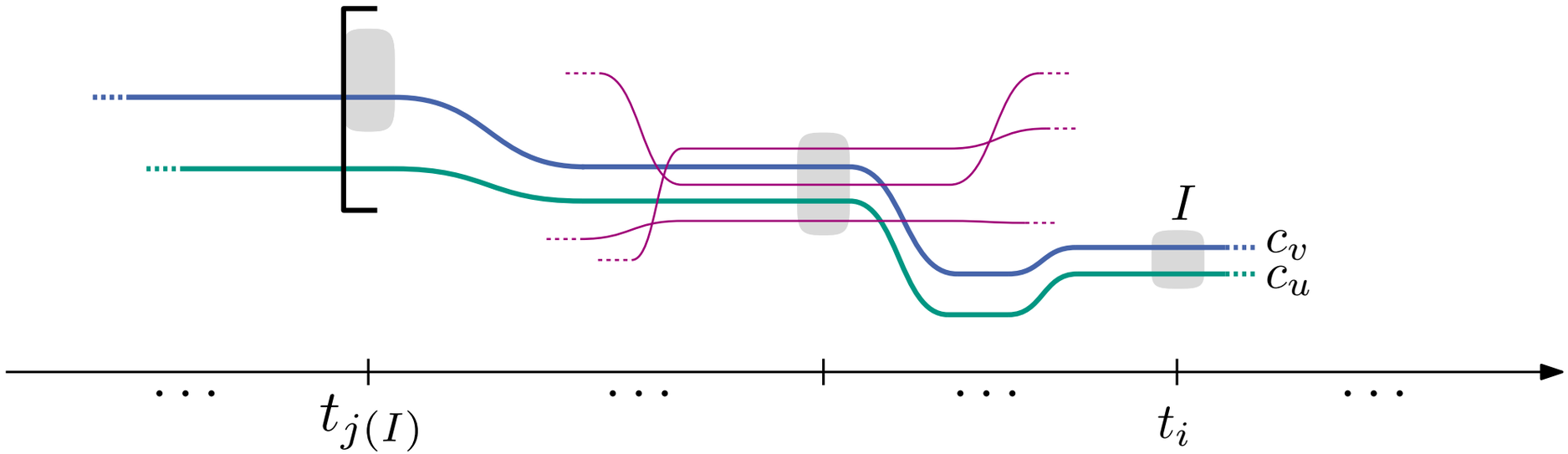
Structural Property 1



Structural Property 1

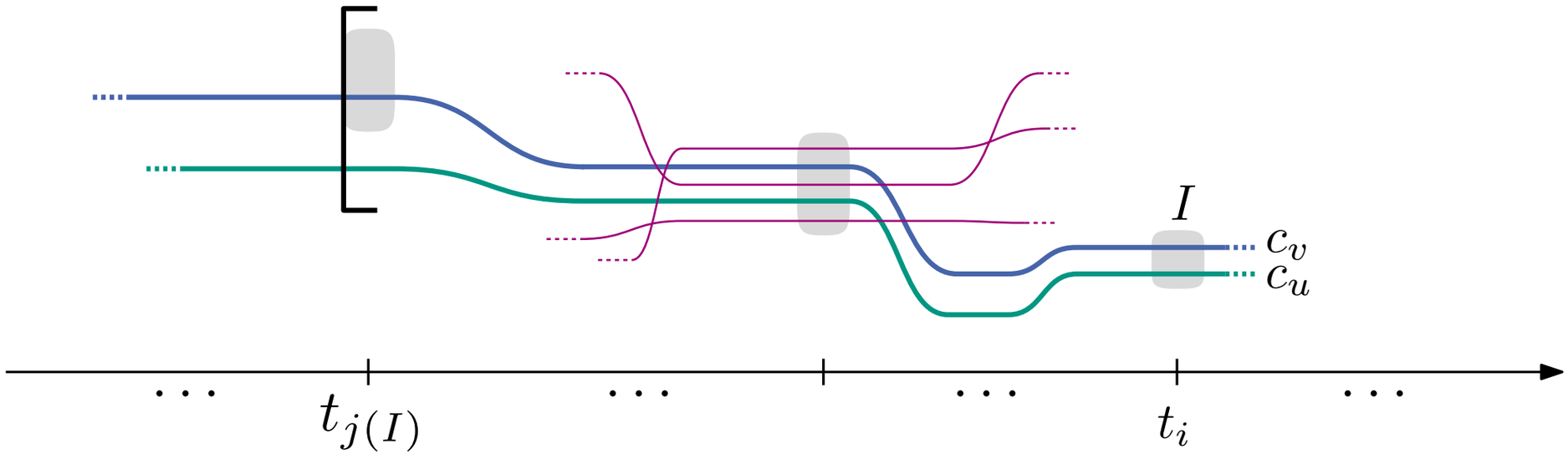


Structural Property 1



Theorem. This untangling does not increase crossings.

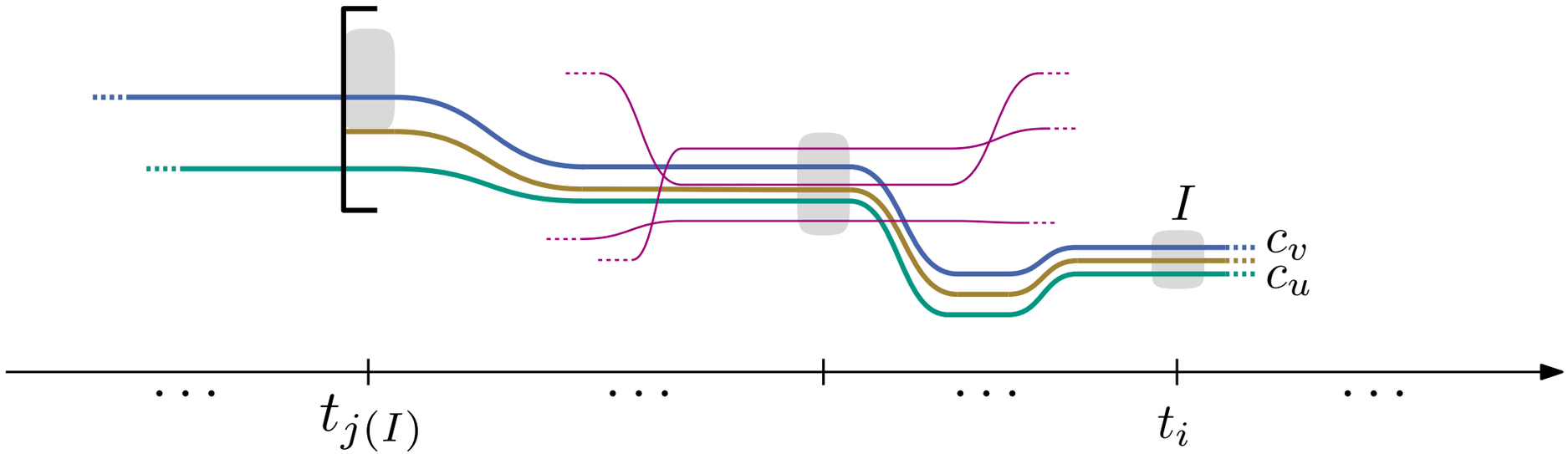
Structural Property 1



Theorem. This untangling does not increase crossings.

$\Rightarrow$ add **symmetry breaking constraints**: $x_{j(I),u,v} = x_{k,u,v}$, $\forall k \in \{j(I)+1, \ldots, i\}$

Structural Property 1

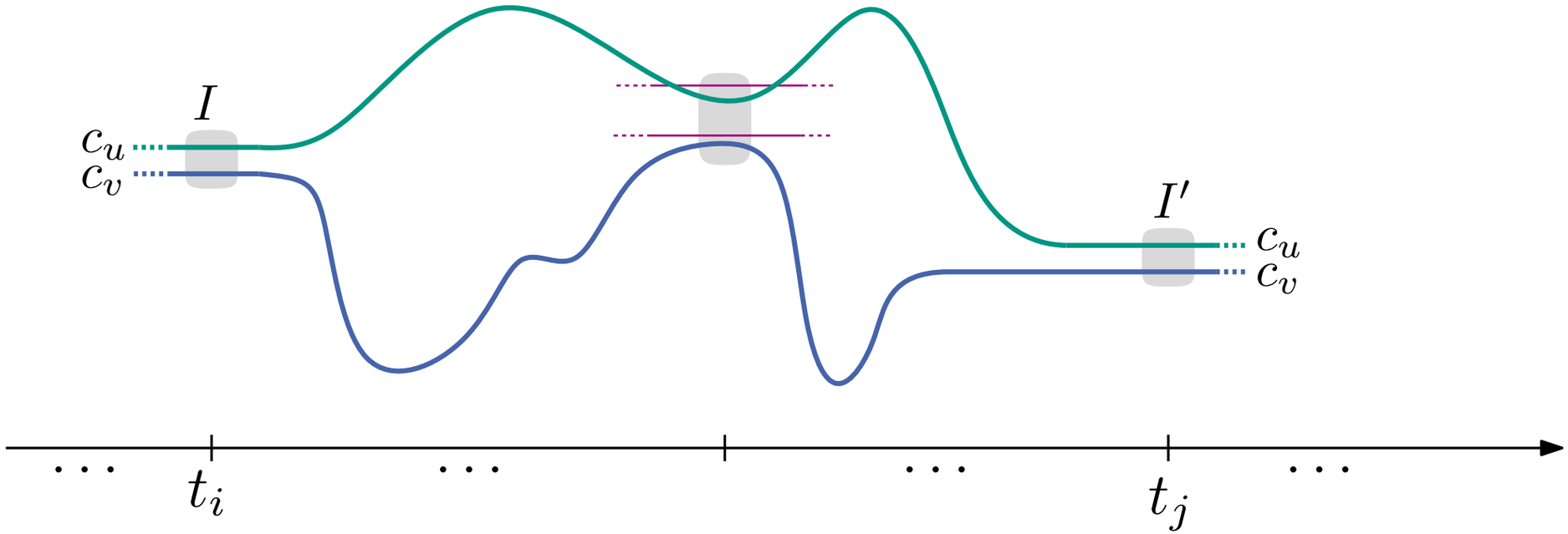


Theorem. This untangling does not increase crossings.

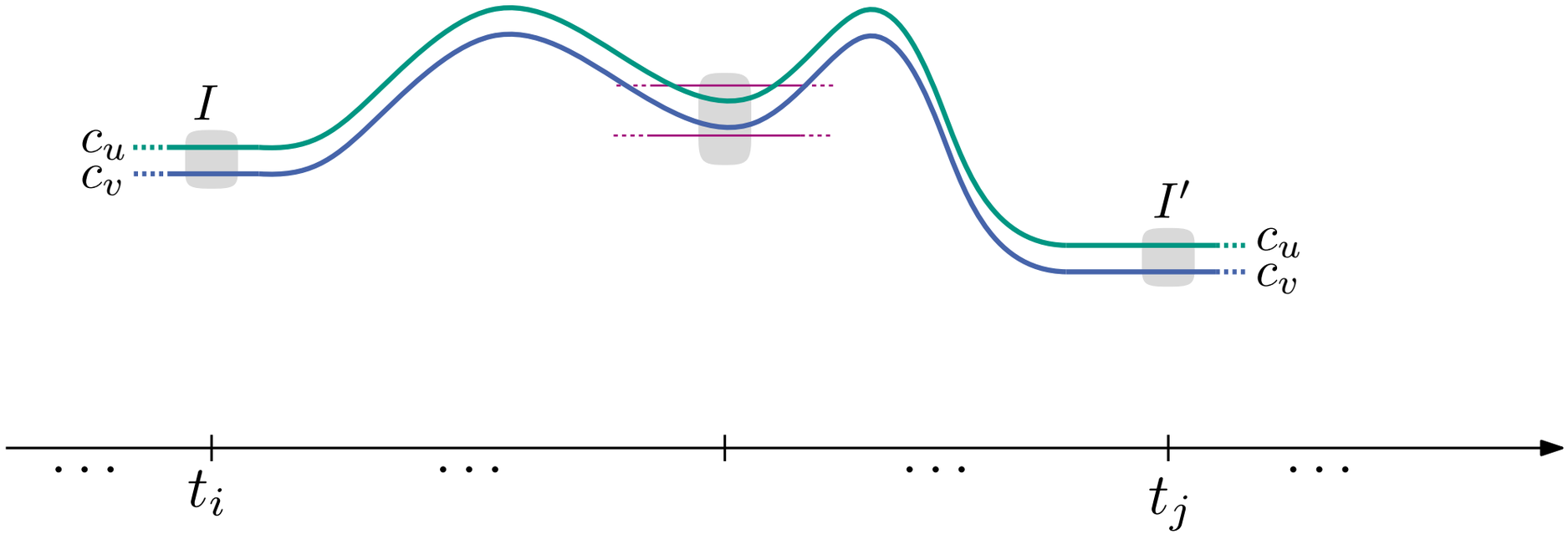
$\Rightarrow$ add **symmetry breaking constraints**: $x_{j(I),u,v} = x_{k,u,v}$, $\forall k \in \{j(I)+1, \ldots, i\}$

This also works for **more than two characters**!

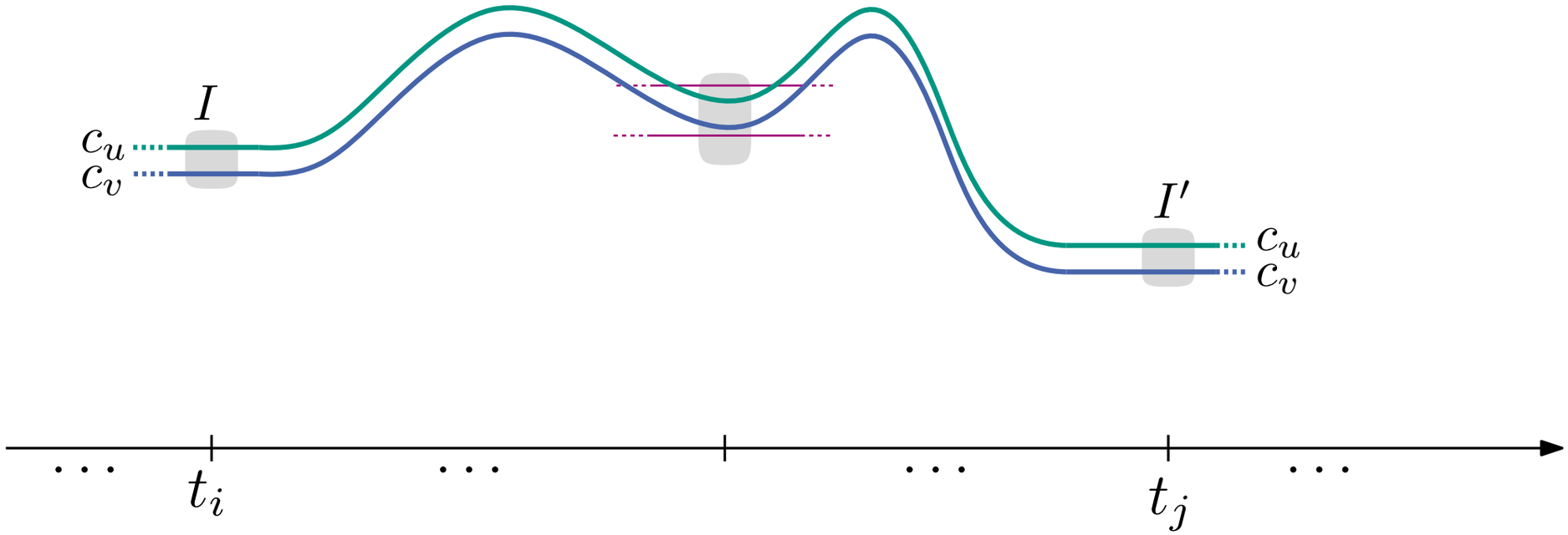
Structural Property 2



Structural Property 2

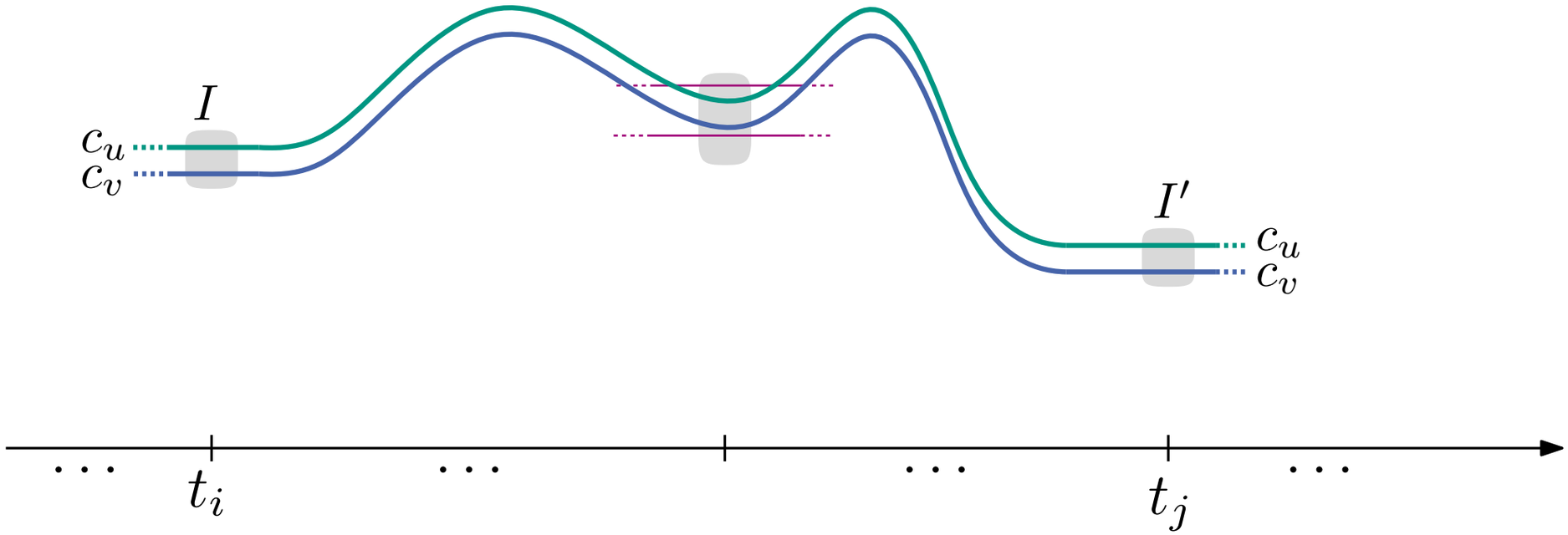


Structural Property 2



$\Rightarrow$ more **symmetry breaking constraints**: $x_{k,u,w} = x_{k,v,w}$,
$\forall k \in \{i+1, \ldots, j-1\}, \forall c_w \in A(k) \setminus \{c_u, c_v\}$

Structural Property 2



$\Rightarrow$ more **symmetry breaking constraints**: $x_{k,u,w} = x_{k,v,w}$, $\forall k \in \{i+1, \ldots, j-1\}, \forall c_w \in A(k) \setminus \{c_u, c_v\}$

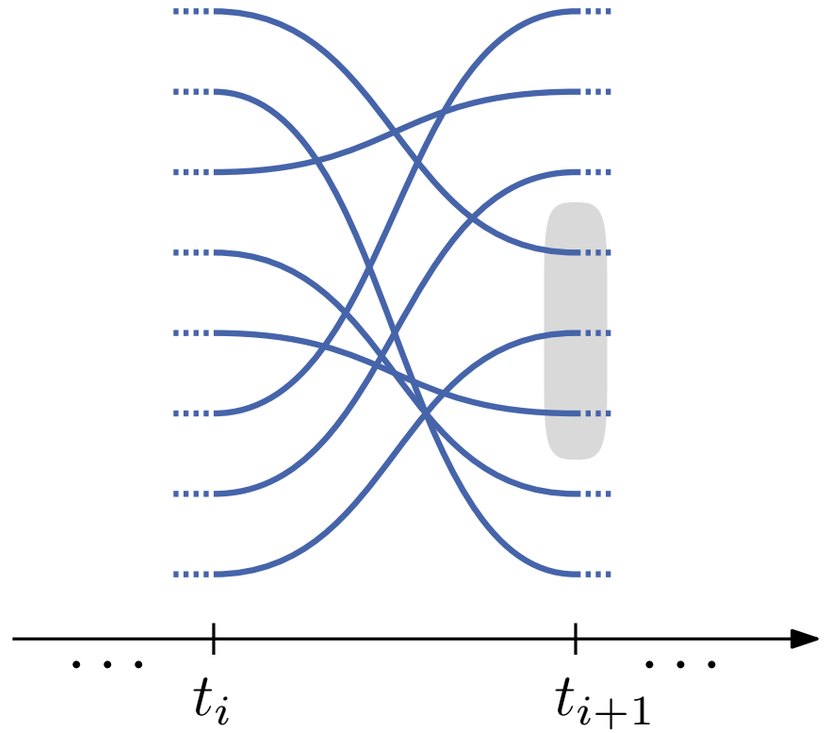
Also **extends** to interaction with **more characters**!

Structural Property 3 and New ILP Model

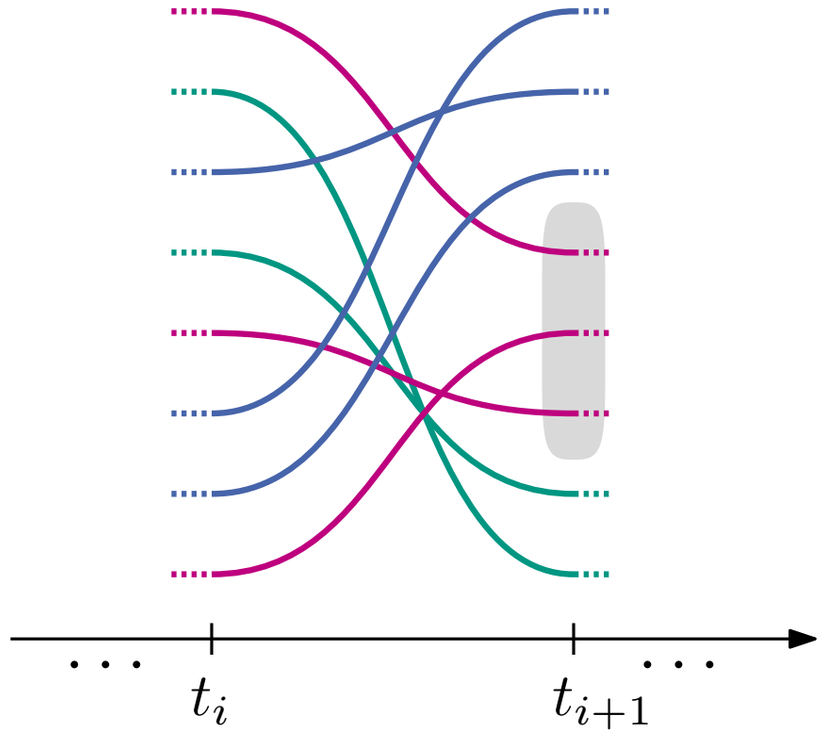
Models have **cubic** number of
transitivity constraints

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2$$
$$\forall i \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i)$$

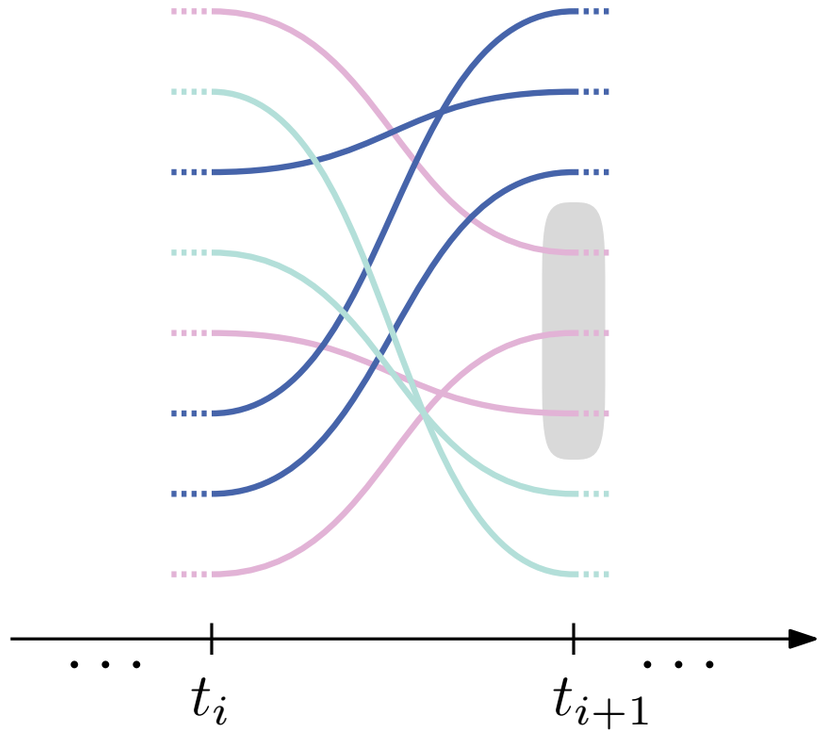
Structural Property 3 and New ILP Model



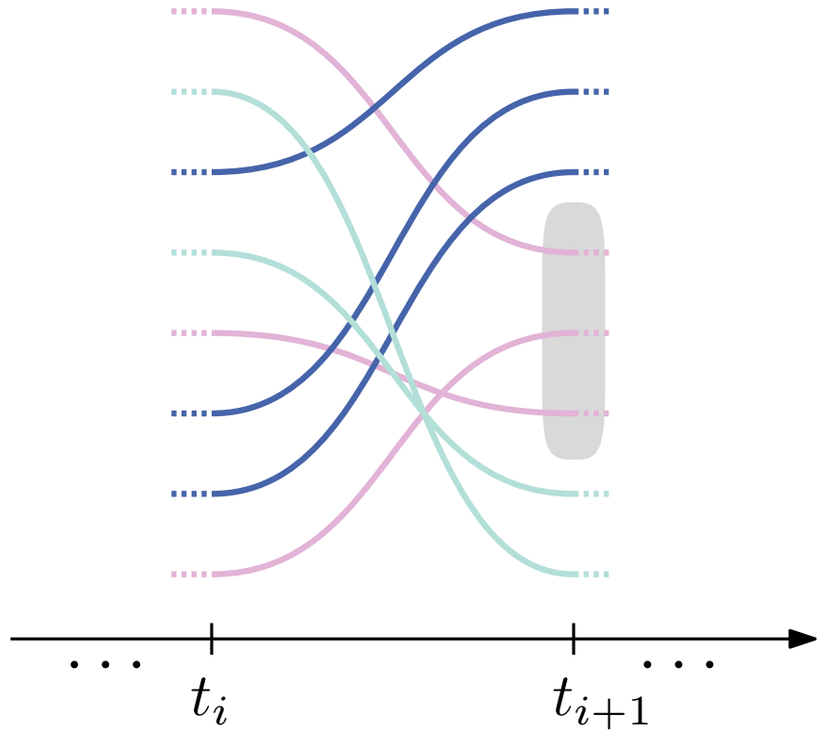
Structural Property 3 and New ILP Model



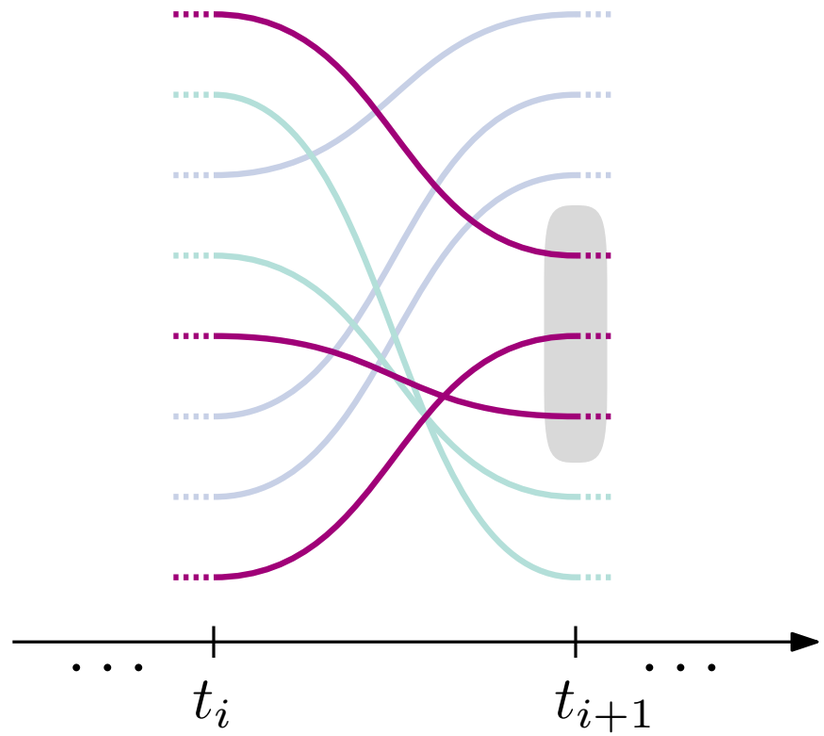
Structural Property 3 and New ILP Model



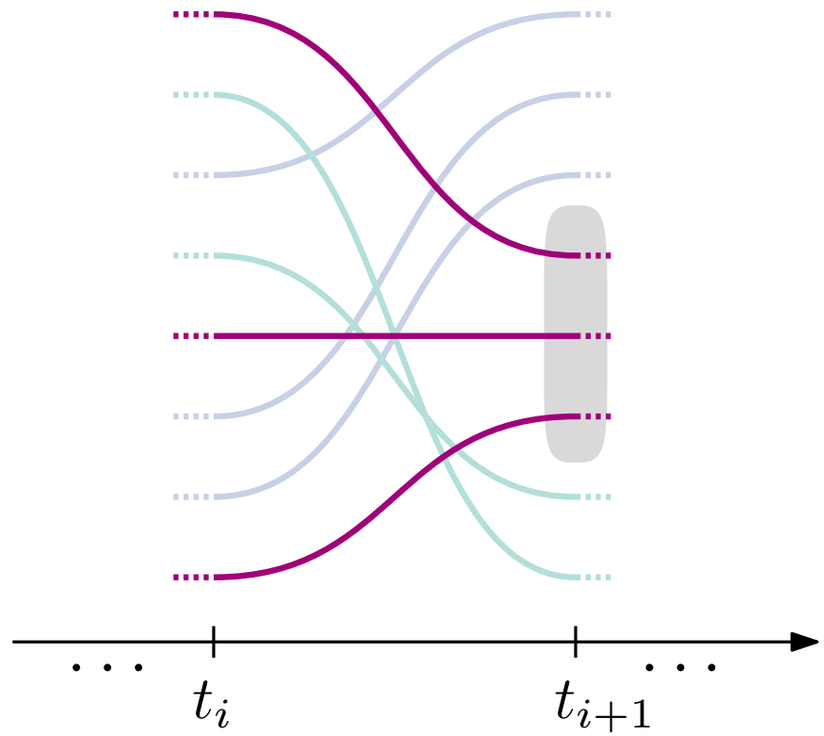
Structural Property 3 and New ILP Model



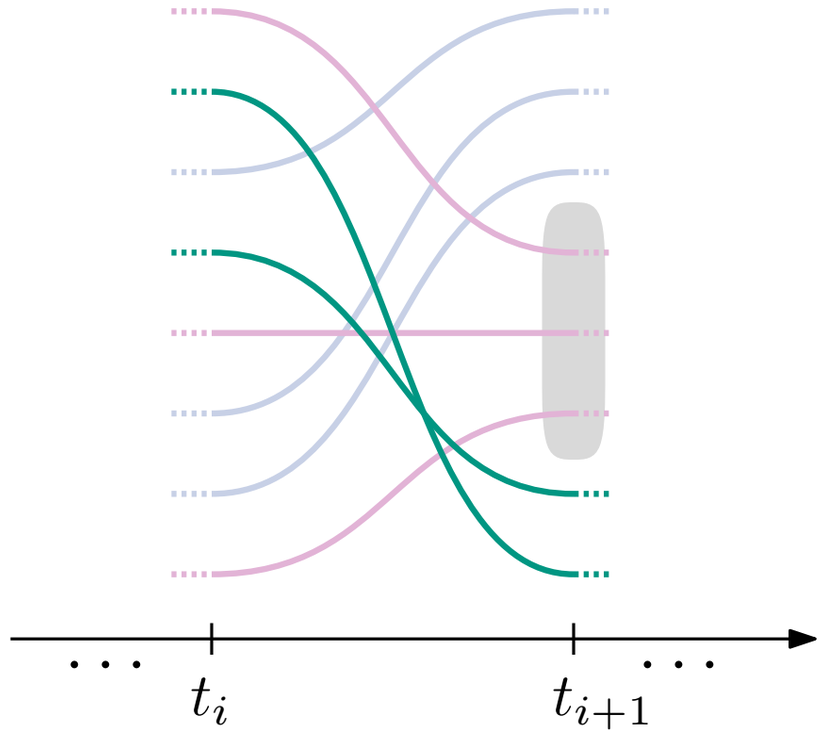
Structural Property 3 and New ILP Model



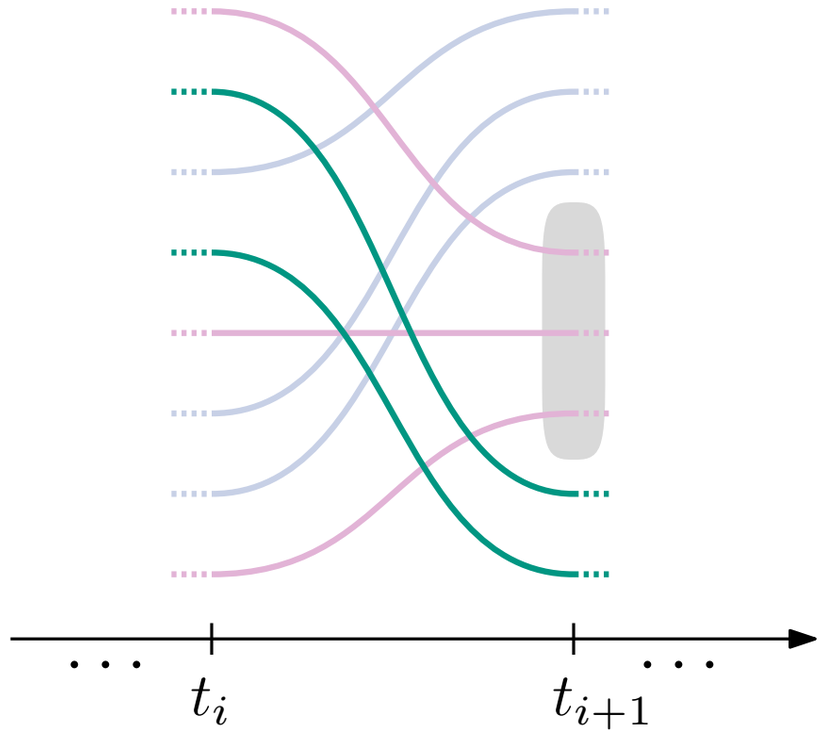
Structural Property 3 and New ILP Model



Structural Property 3 and New ILP Model

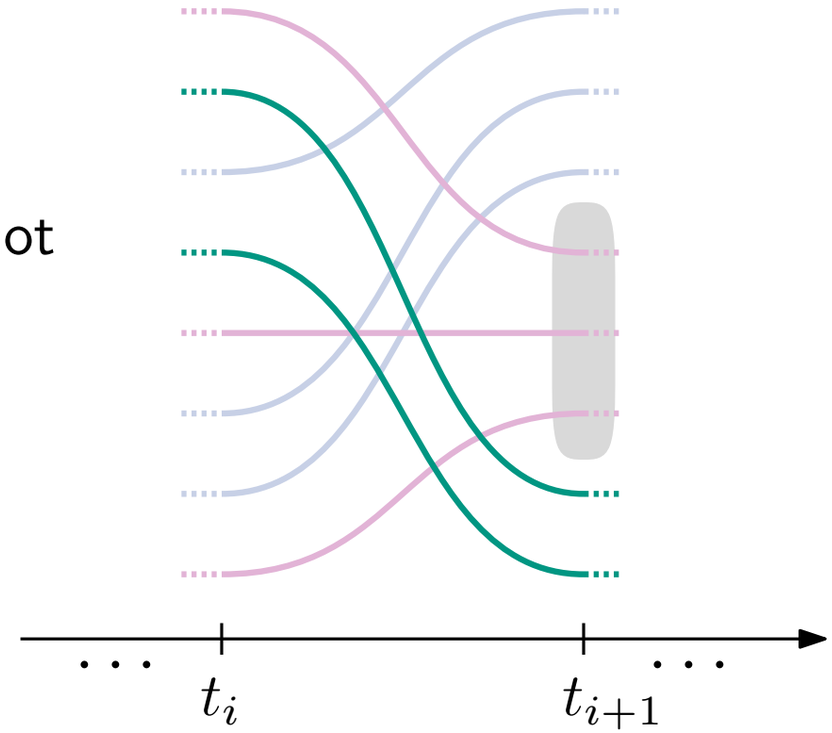


Structural Property 3 and New ILP Model



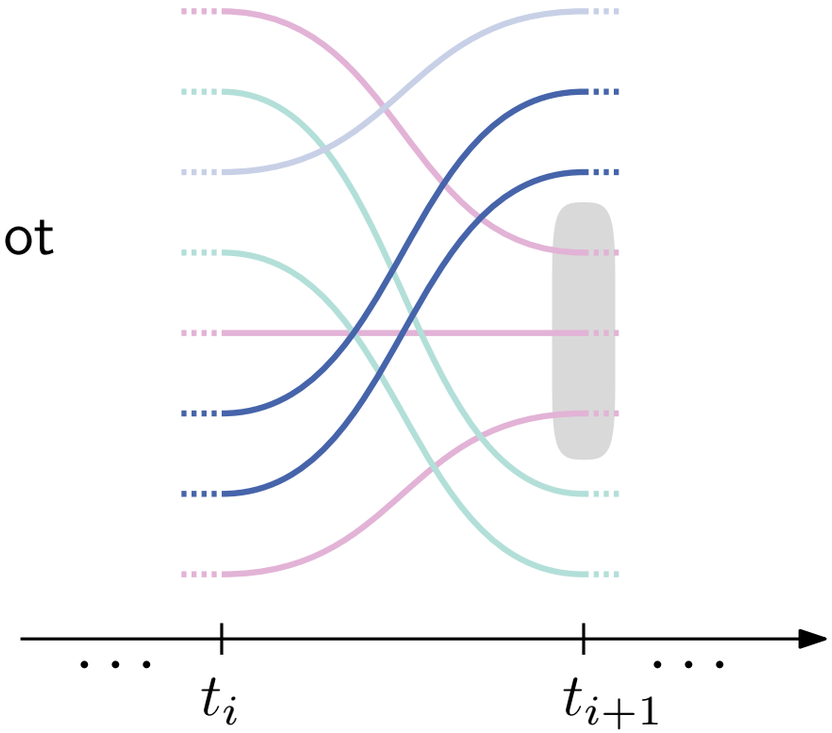
Structural Property 3 and New ILP Model

Theorem. Untangling does not increase crossings.



Structural Property 3 and New ILP Model

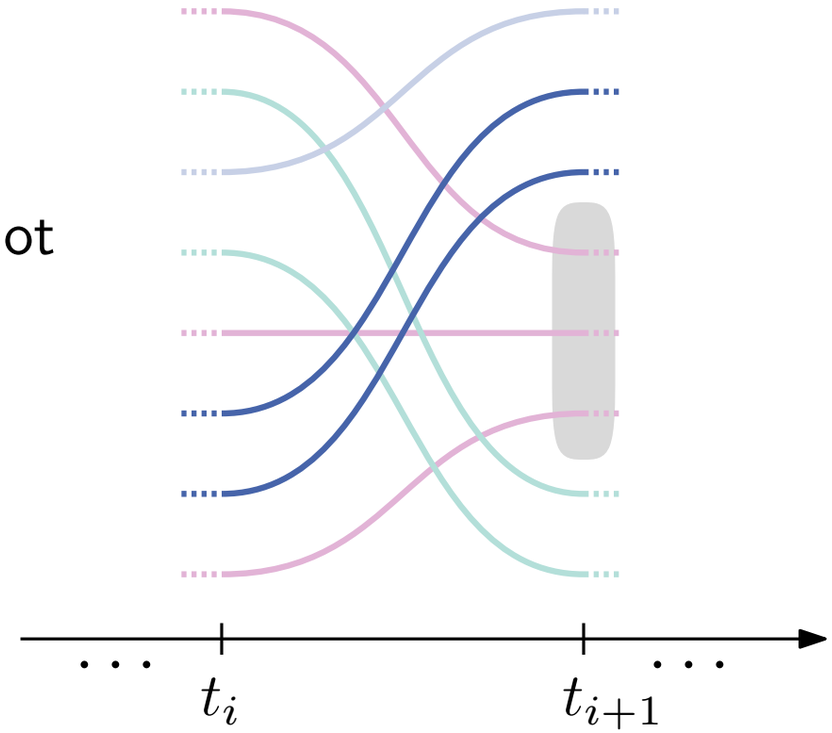
Theorem. Untangling does not increase crossings.



$\Rightarrow$ for a pair of characters above I relative order propagated from t_i

Structural Property 3 and New ILP Model

Theorem. Untangling does not increase crossings.

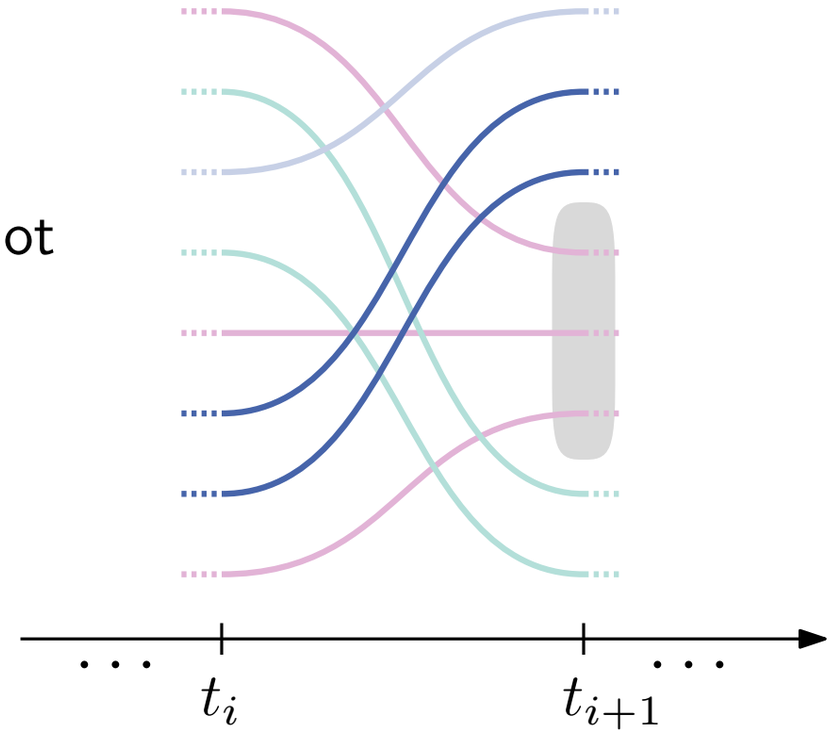


$\Rightarrow$ for a pair of characters above I relative order propagated from t_i

With some **additional** constraints we can get to **quadratic instead of cubic** number of constraints

Structural Property 3 and New ILP Model

Theorem. Untangling does not increase crossings.



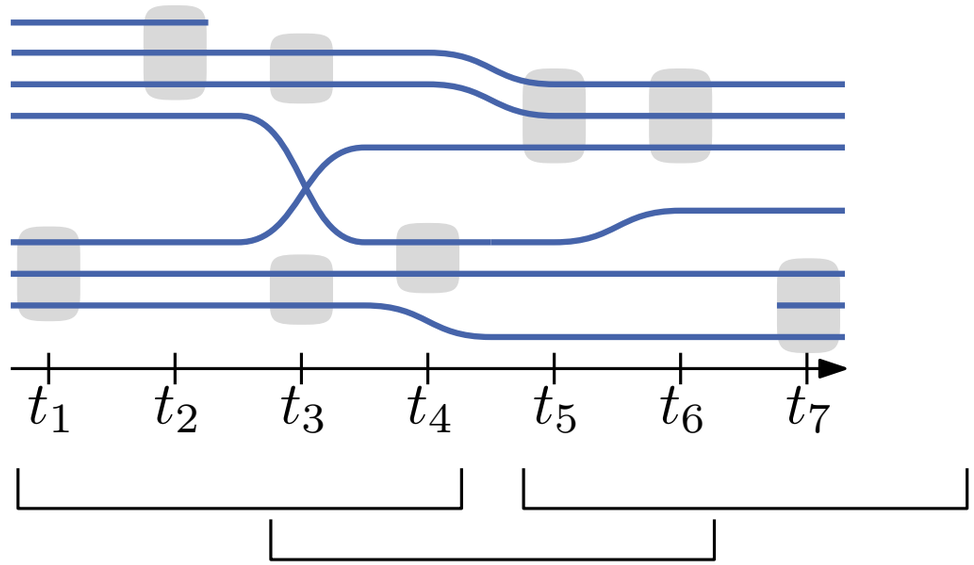
New Model: **PLO**

$\Rightarrow$ for a pair of characters above I relative order propagated from t_i

With some **additional** constraints we can get to **quadratic instead of cubic** number of constraints

Heuristics

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions



Compute optimal solution for **overlapping slices** and **glue them together**

Heuristics

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

ILP solver gives **fractional LP solutions** during solution process

Heuristics

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

ILP solver gives **fractional LP solutions** during solution process

Round them to get good **upper bounds** on optimal solution.

Heuristics

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

ILP solver gives **fractional LP solutions** during solution process

Round them to get good **upper bounds** on optimal solution.

More clever: **propagate** order from t_i to t_{i+1} if binary variables close to 0.5

Heuristics

- **structural** problem-specific **insights**,
- **initial heuristic**, and
- **exploitation heuristics** for fractional solutions

ILP solver gives **fractional LP solutions** during solution process

Round them to get good **upper bounds** on optimal solution.

More clever: **propagate** order from t_i to t_{i+1} if binary variables close to 0.5

Afterwards: apply **local search** based on **structural observations** and a **barycenter-type heuristic**

Experiments

Algorithms

- **MC**: Reimplemented Max-Cut Formulation of [GJLM, 2016]
- **LIN**: Linearized model
- **QDR**: Quadratic model
- **PLO**: New model based on structural insights

Experiments

Algorithms

- **MC**: Reimplemented Max-Cut Formulation of [GJLM, 2016]
- **LIN**: Linearized model
- **QDR**: Quadratic model
- **PLO**: New model based on structural insights

LIN, **QDR**, and **PLO** are enriched by symmetry breaking constraints (**SBC**), initial heuristic (**INIT**), and rounding (**RND**). Transitivity constraints are added **lazily**

Experiments

Algorithms

- **MC**: Reimplemented Max-Cut Formulation of [GJLM, 2016]
- **LIN**: Linearized model
- **QDR**: Quadratic model
- **PLO**: New model based on structural insights

LIN, **QDR**, and **PLO** are enriched by symmetry breaking constraints (**SBC**), initial heuristic (**INIT**), and rounding (**RND**). Transitivity constraints are added **lazily**

All algorithms use **Gurobi** (Gurobi allows quadratic models)

Instances

59 instances: **books**, **movies**, **publication data**

Instances

59 instances: **books**, **movies**, **publication data**

- 24 instances too easy
- 12 instances too hard

Instances

59 instances: **books**, **movies**, **publication data**

- 24 instances too easy
- 12 instances too hard

Evaluation: **23** instances

Instances

59 instances: **books**, **movies**, **publication data**

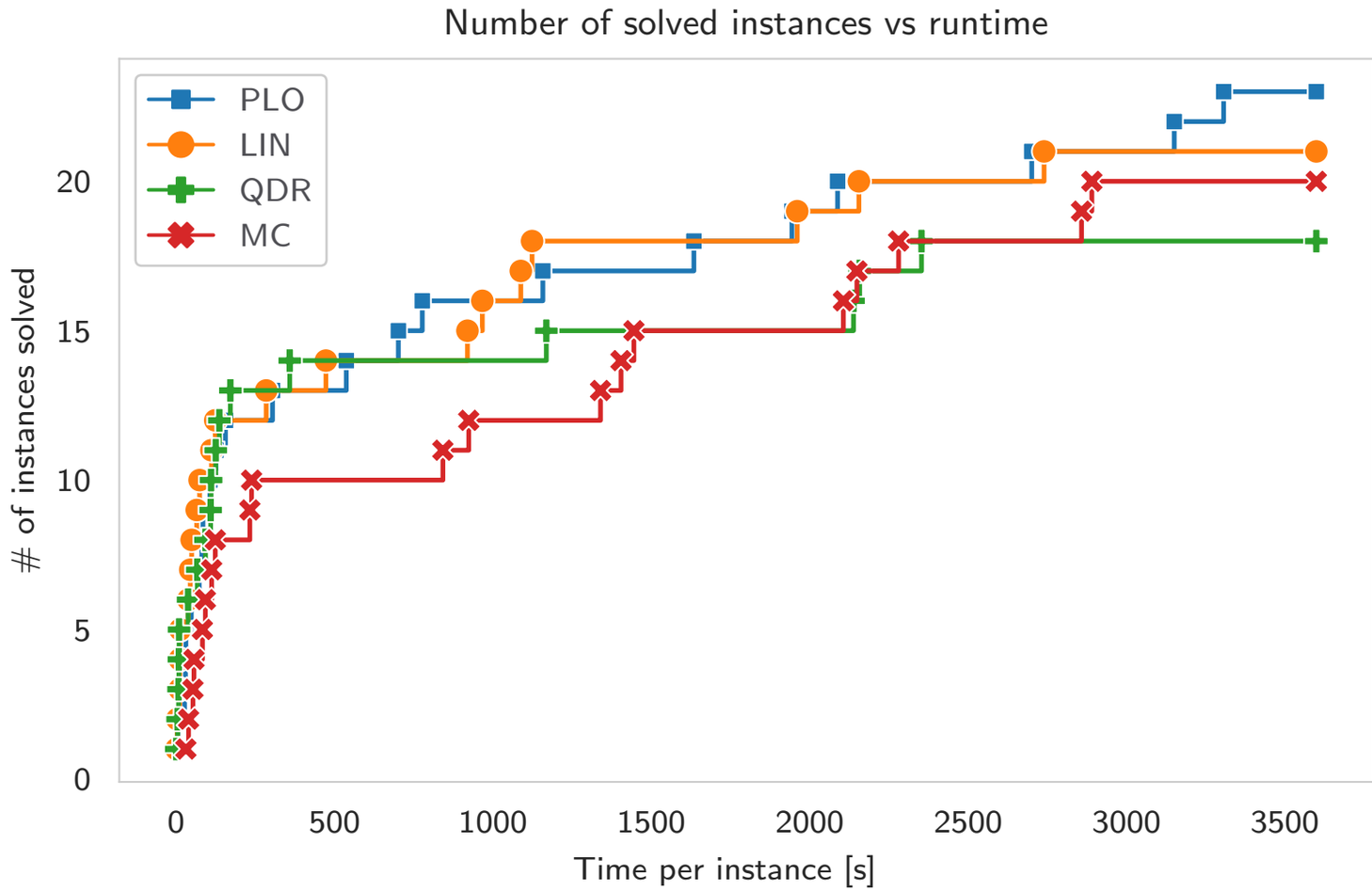
- 24 instances too easy
- 12 instances too hard

Evaluation: **23** instances

35–88 characters, 54–329 interactions

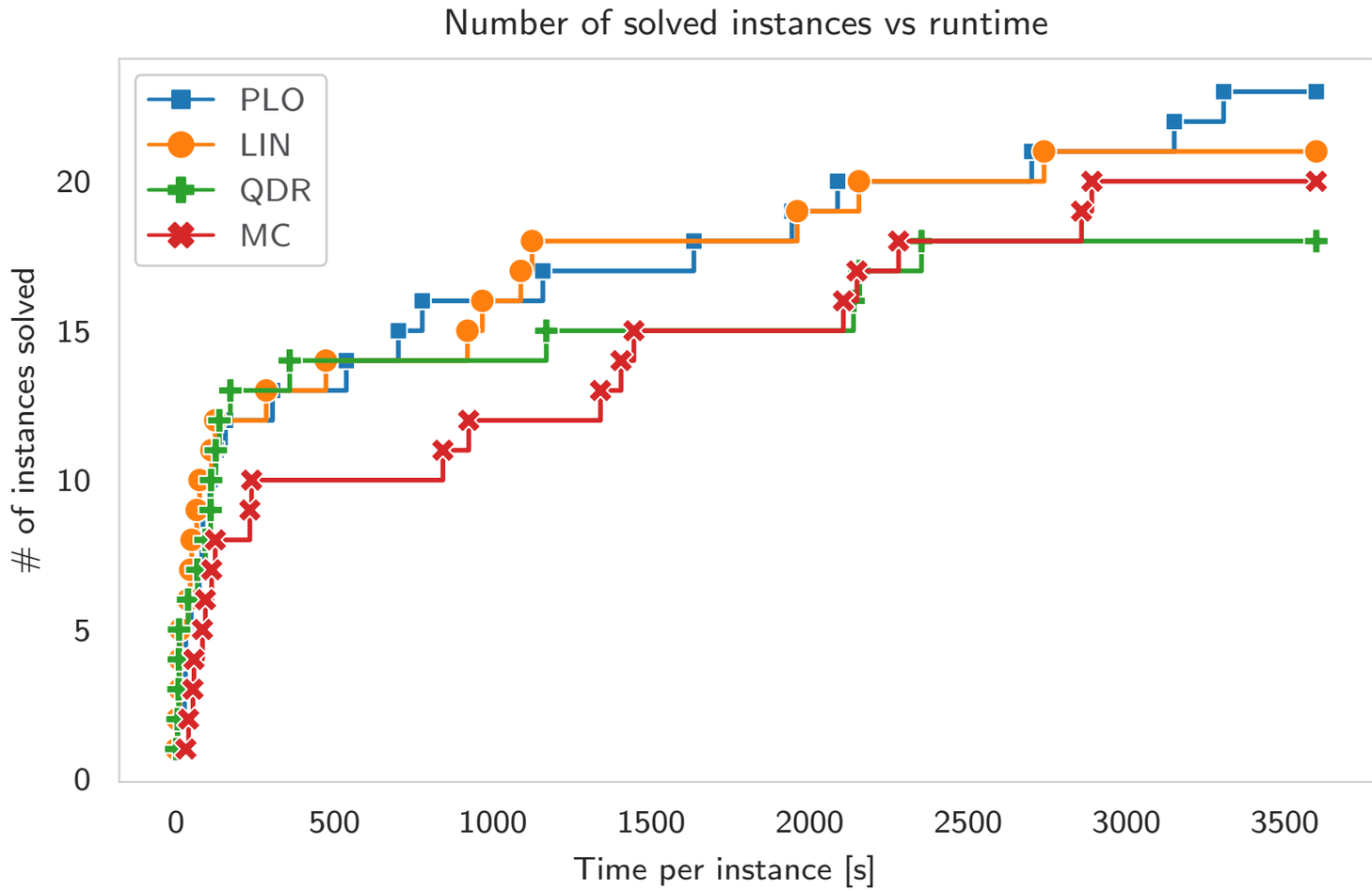
Results

Results



Timeout: $3600s$

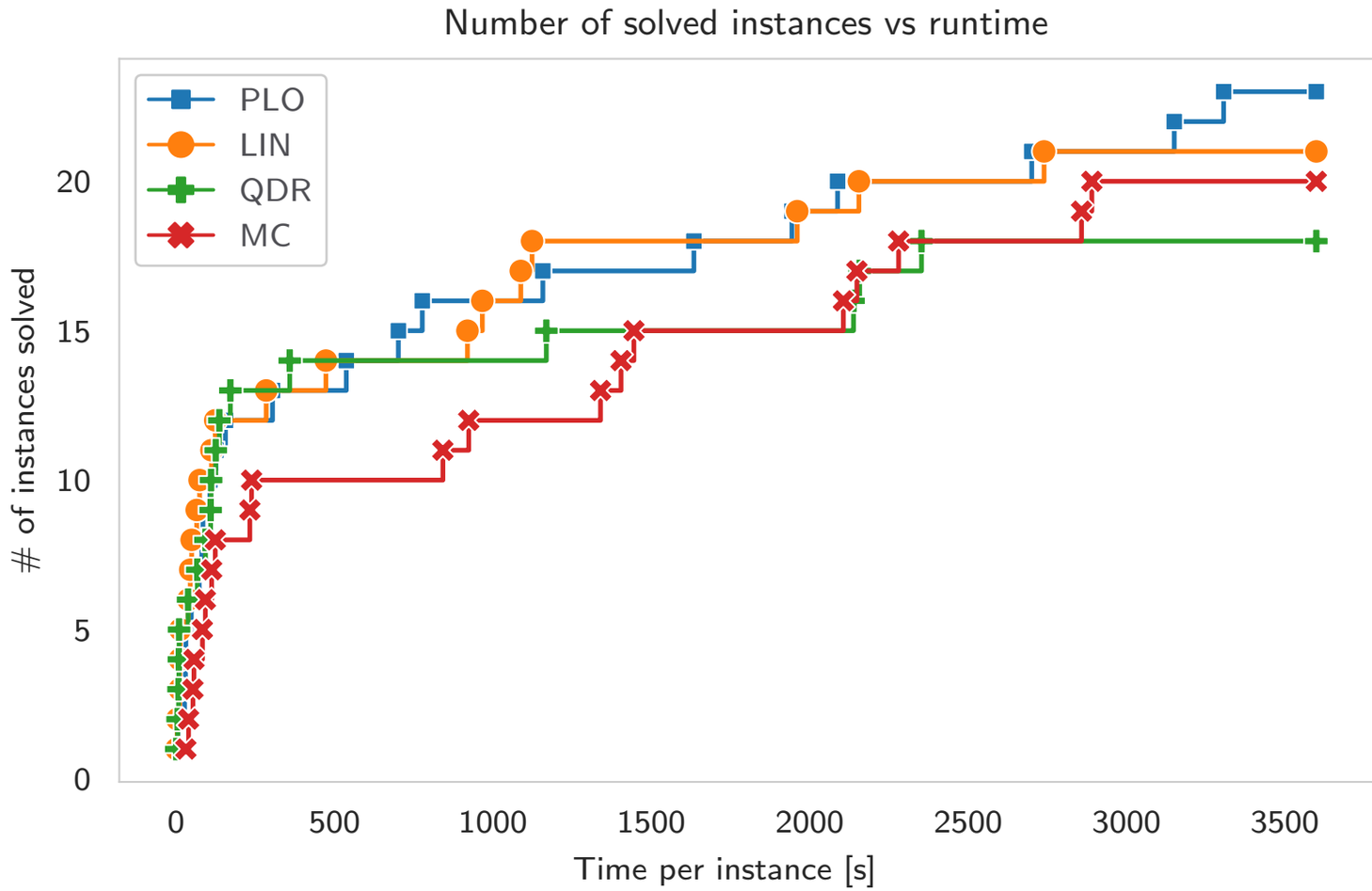
Results



Timeout: $3600s$

New model solves all 23 considered instances within time limit!

Results

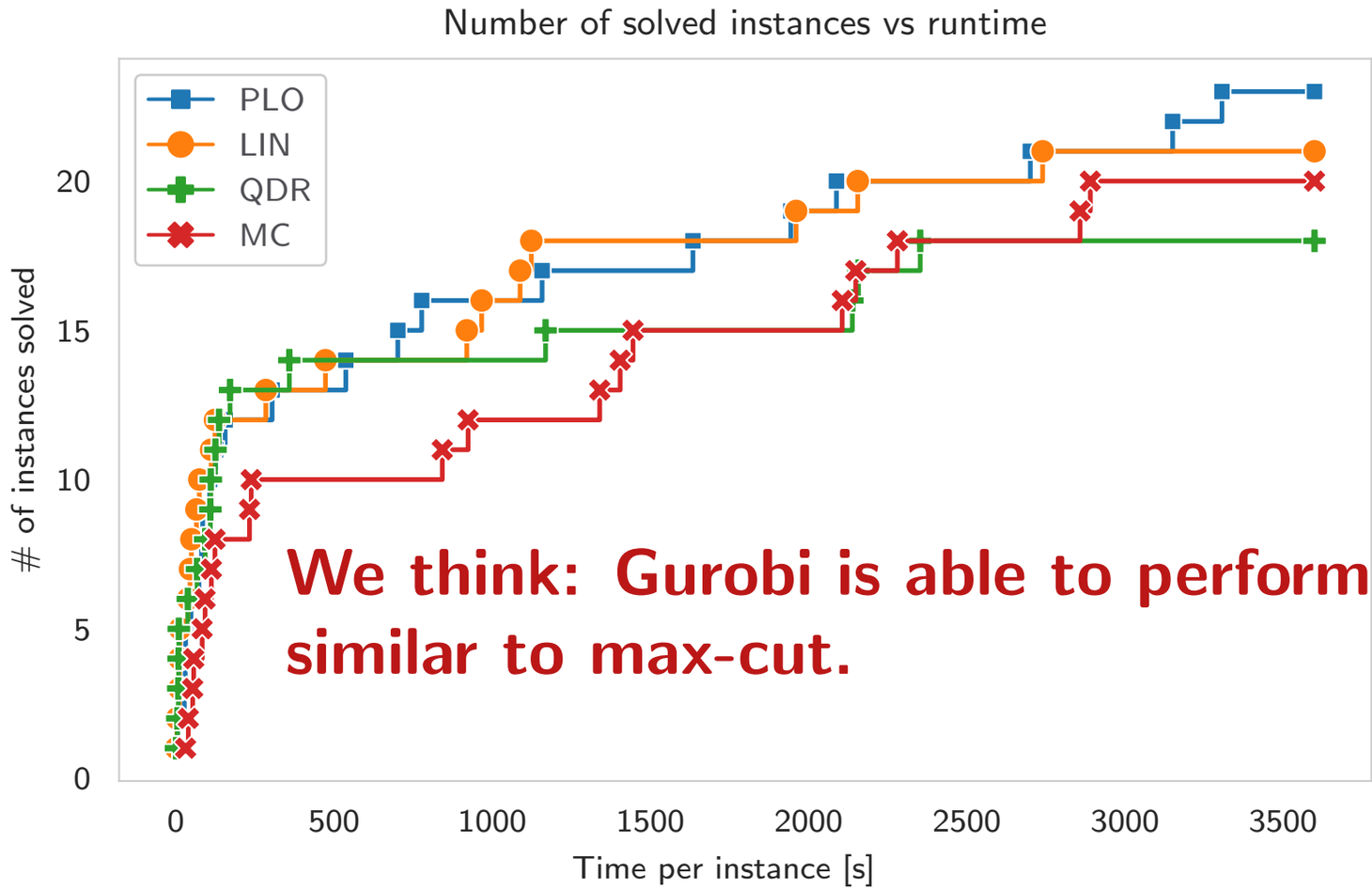


Timeout: $3600s$

New model solves all 23 considered instances within time limit!

LIN and **PLO** perform better than **MC** by [GJLM, 2016], **2–3 times faster**

Results



Timeout: $3600s$

We think: Gurobi is able to perform transformation similar to max-cut.

New model solves all 23 considered instances within time limit!

LIN and **PLO** perform better than **MC** by [GJLM, 2016], **2–3 times faster**

Ablation Study

Ablation study: Disable one component at a time to discern its effect.

Ablation Study

Ablation study: Disable one component at a time to discern its effect.

Components:

- **SBC**: symmetry breaking constraints
- **INIT**: initial heuristic
- **RND**: exploitation of fractional solutions

Ablation Study

Ablation study: Disable one component at a time to discern its effect.

Components:

- **SBC**: symmetry breaking constraints
- **INIT**: initial heuristic
- **RND**: exploitation of fractional solutions

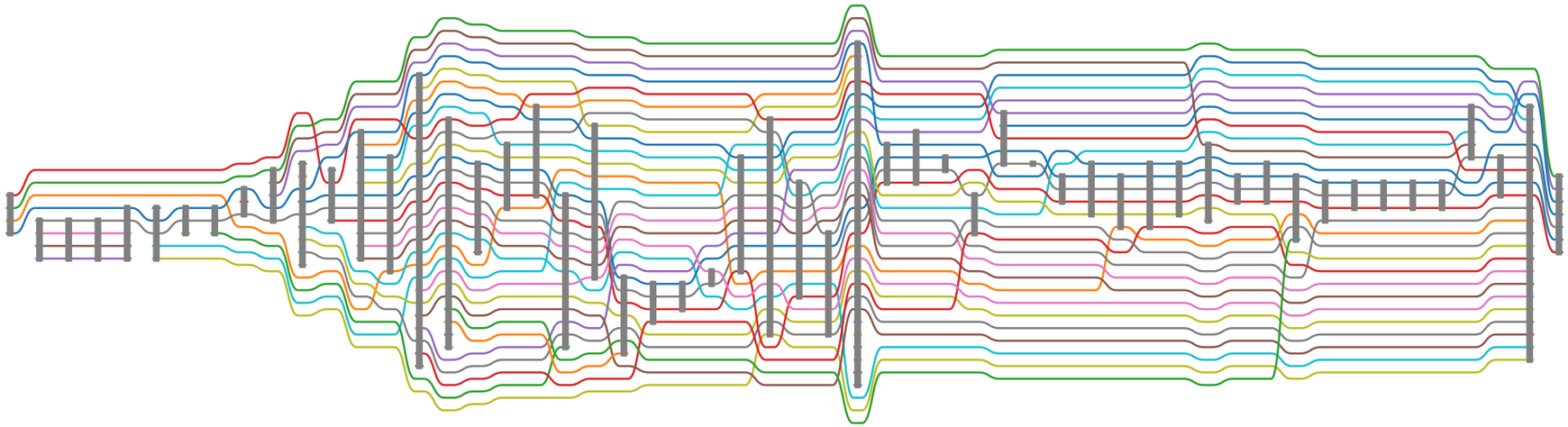
speedup factor	**PLO**	**LIN**	**QDR**
SBC	1.12	1.35	1.42
INIT	1.05	1.09	0.97
RND	1.50	1.37	1.52

Conclusion

Enriching **simpler models** by problem-specific insights can **outperform more complex models** (at least when implemented in Gurobi).

Conclusion

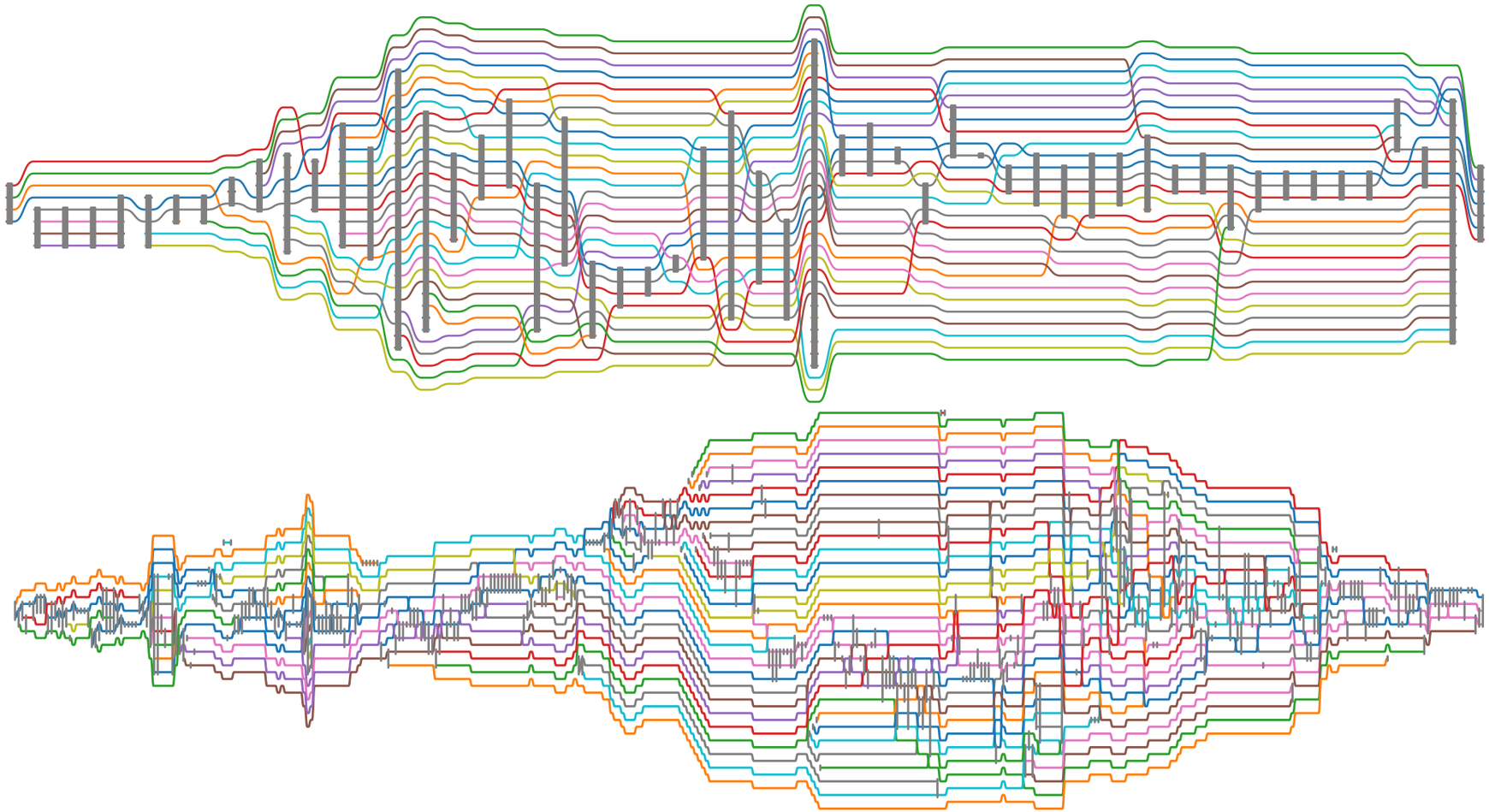
Enriching **simpler models** by problem-specific insights can **outperform more complex models** (at least when implemented in Gurobi).



Harry Potter 1
PLO: 776s
LIN: 1123s
MC: 1444s
QDR: timeout

Conclusion

Enriching **simpler models** by problem-specific insights can **outperform more complex models** (at least when implemented in Gurobi).

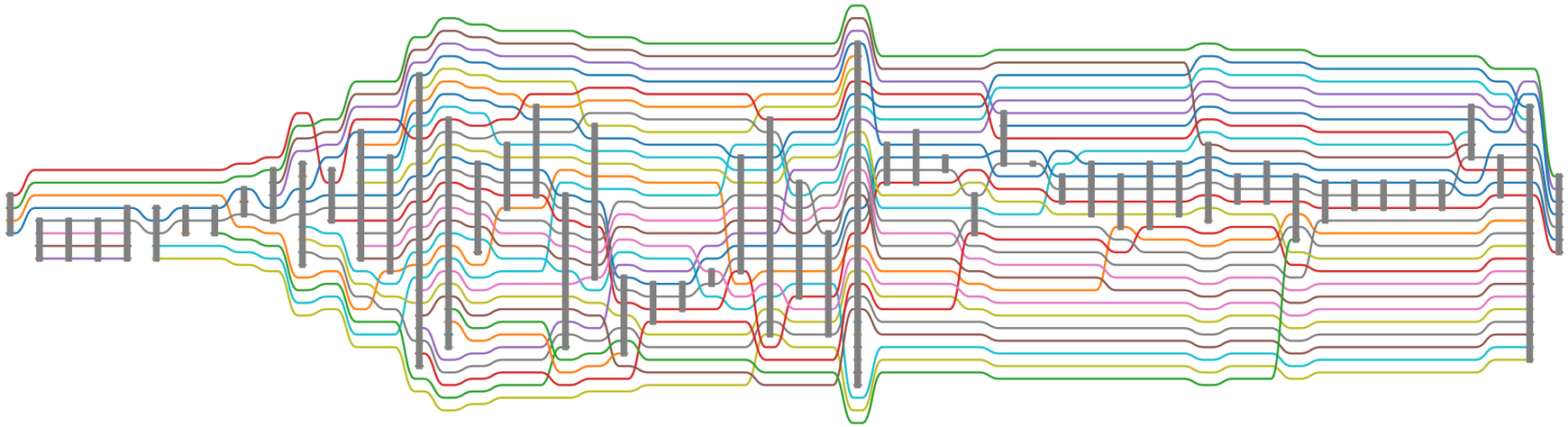


Harry Potter 1
PLO: 776s
LIN: 1123s
MC: 1444s
QDR: timeout

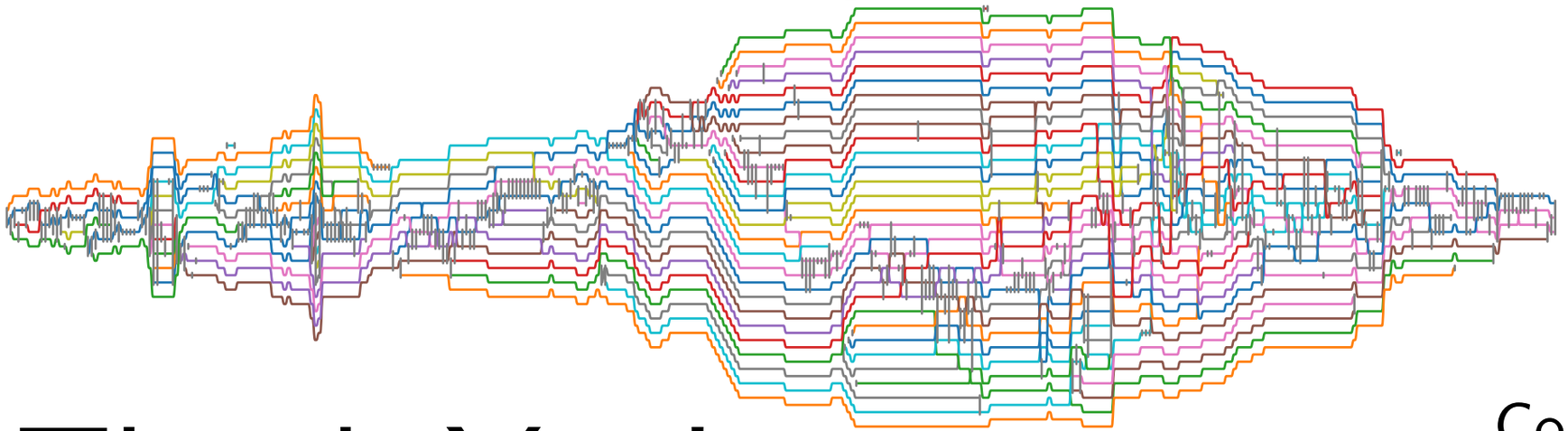
Les Misérables
PLO: $\approx$7h

Conclusion

Enriching **simpler models** by problem-specific insights can **outperform more complex models** (at least when implemented in Gurobi).



Harry Potter 1
PLO: 776s
LIN: 1123s
MC: 1444s
QDR: timeout



Les Misérables
PLO: $\approx 7h$

Thank You!

Code and instances

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} x_{i,u,v} x_{i+1,v,u} \qquad \text{(QDR)}$$

$$x_{i,u,v} = 1 - x_{i,v,u} \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \qquad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$

$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

$\mathrm{AC}(t_i, t_{i+1})$ . . . characters active at t_i and t_{i+1}

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} x_{i,u,v} x_{i+1,v,u} \qquad \text{(QDR)}$$

$$x_{i,u,v} = 1 - x_{i,v,u} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \quad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$
$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

$\mathrm{AC}(t_i, t_{i+1})$ . . . characters active at t_i and t_{i+1}

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} x_{i,u,v} x_{i+1,v,u} \qquad \text{(QDR)}$$

Symmetry

$$x_{i,u,v} = 1 - x_{i,v,u} \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \qquad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$
$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \qquad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

$\mathrm{AC}(t_i, t_{i+1})$ . . . characters active at t_i and t_{i+1}

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} x_{i,u,v} x_{i+1,v,u} \qquad \text{(QDR)}$$

Transitivity

$$x_{i,u,v} = 1 - x_{i,v,u} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \quad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$
$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

$\mathrm{AC}(t_i, t_{i+1})$ . . . characters active at t_i and t_{i+1}

A Simple Quadratic Integer Program

Binary variable $x_{i,u,v}$ which is 1 iff character c_u is before c_v in π_i

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} x_{i,u,v} x_{i+1,v,u} \qquad \text{(QDR)}$$

$$x_{i,u,v} = 1 - x_{i,v,u} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

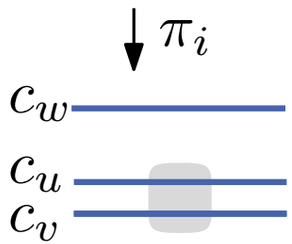
$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \quad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$

$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

Tree contraints



$\mathrm{AC}(t_i, t_{i+1})$ . . . characters active at t_i and t_{i+1}

Linearized Model

linearized objective

$$\min \sum_{i=1}^{\ell-1} \sum_{c_u, c_v \in \mathrm{AC}(t_i, t_{i+1})} y_{i,u,v} \qquad \text{(LIN)}$$

$$y_{i,u,v} \geq x_{i,u,v} - x_{i+1,u,v} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i, t_{i+1}) \qquad \text{(CR)}$$

$$x_{i,u,v} = 1 - x_{i,v,u} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i) \text{ with } u < v \qquad \text{(EQ)}$$

$$x_{i,u,v} + x_{i,v,w} + x_{i,w,u} \leq 2 \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v, c_w \in \mathrm{AC}(t_i) \qquad \text{(LOP)}$$

$$x_{i,u,w} = x_{i,v,w} \quad \text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \qquad \text{(TREE)}$$
$$c_u, c_v \in \mathrm{char}(I), u < v; c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I)$$

$$x_{i,u,v} \in \{0, 1\} \quad \text{for all } i = 1, \ldots, \ell; c_u, c_v \in \mathrm{AC}(t_i), \qquad \text{(BIN)}$$

Linear Max-Cut Model by [Gronemann et al., 2016]

Gronemann et al. reformulated the problem as a **Max-Cut** problem on specific weighted graph G and computed optimal solutions on real-world storylines

exponential number of constraints
$\rightarrow$ branch and cut, complicated to implement

$$\max \sum_{e \in E_M} w_e z_e \qquad \text{(CUT)}$$

$$\sum_{e \in F} z_e - \sum_{e \in C \setminus F} z_e \leq |F| - 1 \quad \text{for all cycles } C \subseteq E_M, F \subseteq C, |F| \text{ odd} \qquad \text{(CYC)}$$

$$0 \leq z_{(v^*, c^i_{uv})} + z_{(v^*, c^i_{vw})} - z_{(v^*, c^i_{uw})} \leq 1 \quad \text{for all } i = 1, \ldots, \ell; c^i_{uv}, c^i_{vw}, c^i_{uw} \in V_i \text{ with } c_u \prec_{\hat{\pi}_i} c_v \prec_{\hat{\pi}_i} c_w \qquad \text{(LOPC)}$$

$$\left.\begin{aligned} z_{(v^*, c^i_{uw})} = z_{(v^*, c^i_{vw})} \text{ if } c_u, c_v \prec_{\hat{\pi}_i} c_w \\ z_{(v^*, c^i_{wu})} = z_{(v^*, c^i_{wv})} \text{ if } c_u, c_v \succ_{\hat{\pi}_i} c_w \end{aligned}\right\} \quad \begin{aligned} &\text{for all } i = 1, \ldots, \ell; I \in \mathcal{I}(t_i); \\ &c_u, c_v \in \mathrm{char}(I); c_w \in \mathrm{AC}(t_i) \setminus \mathrm{char}(I) \end{aligned} \qquad \text{(TRC)}$$

$$z_e \in \{0, 1\} \quad \text{for all } e \in E_M \qquad \text{(BIC)}$$

More Results