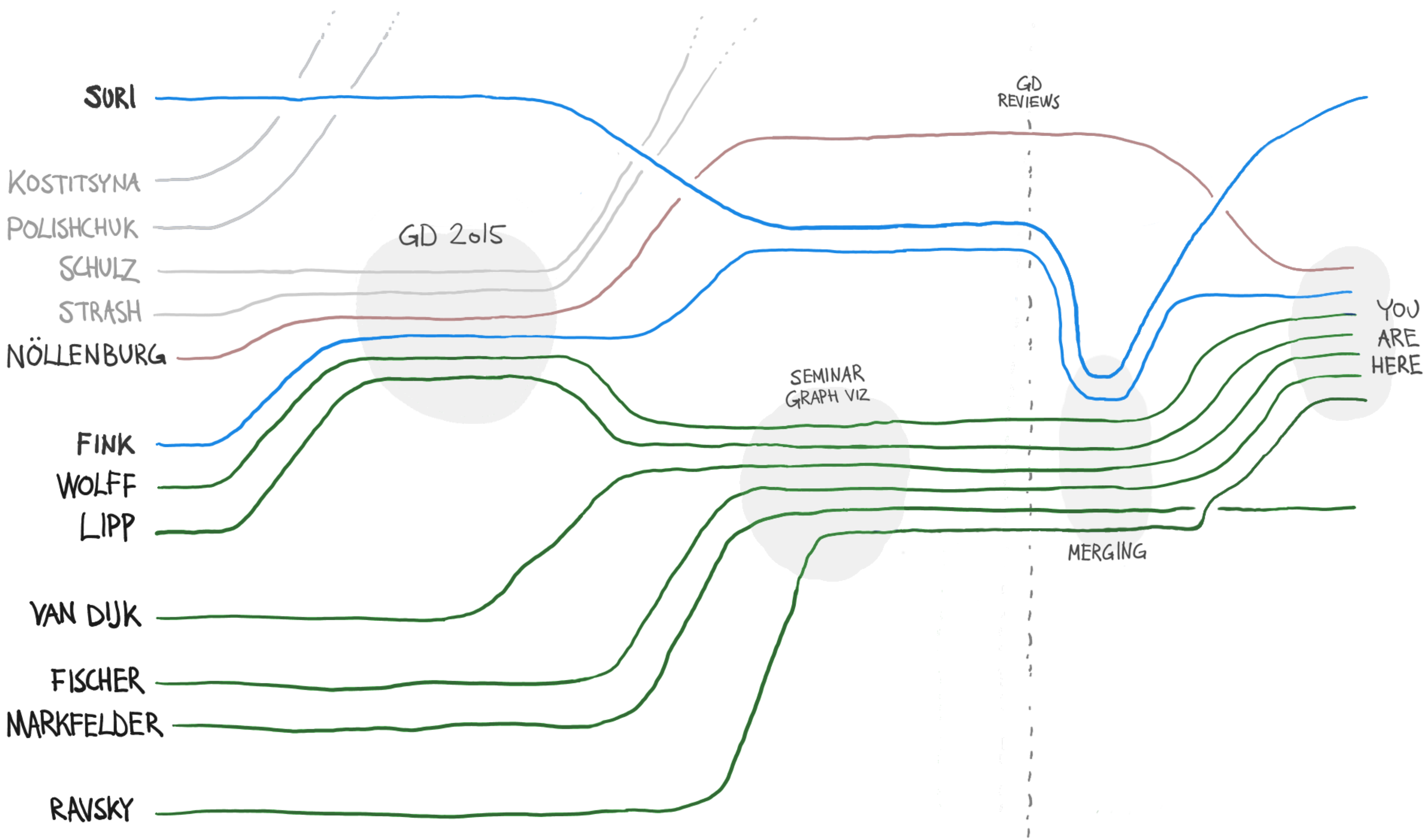
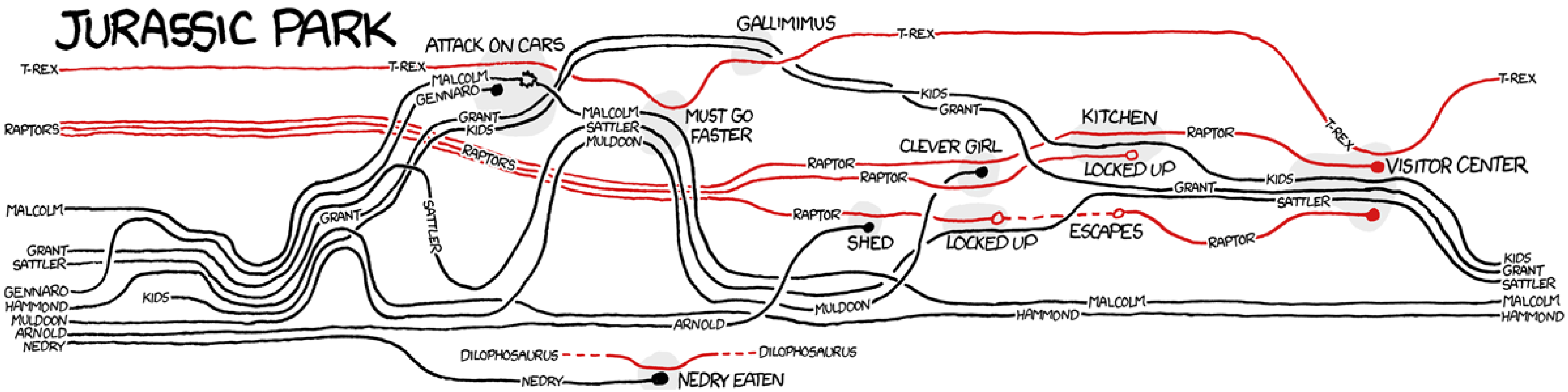
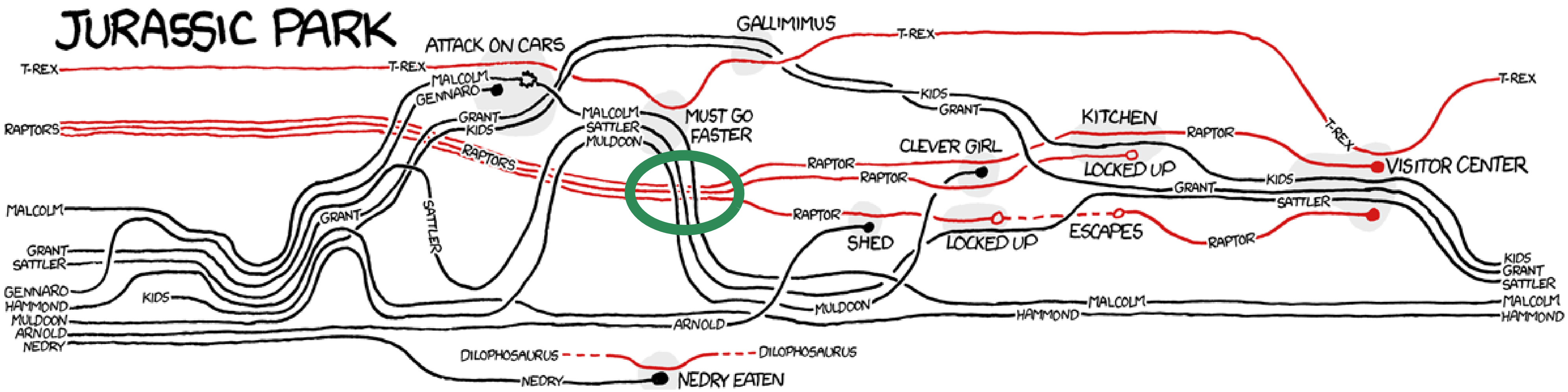
Block Crossings in Storyline Visualizations

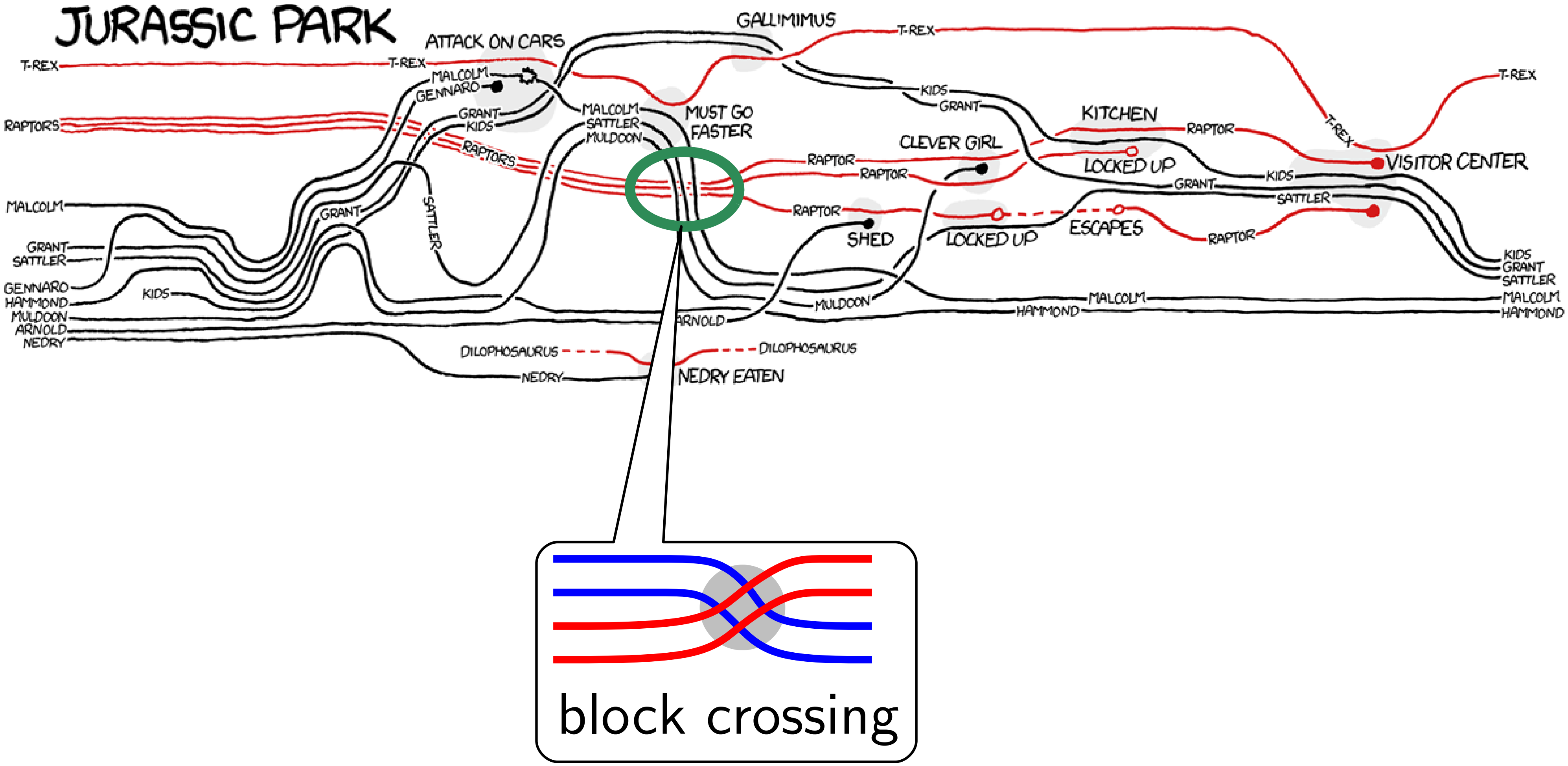
Thomas van Dijk, *Martin Fink*, Norbert Fischer, Fabian Lipp, Peter Markfelder, Alexander Ravsky, Subhash Suri, and Alexander Wolff

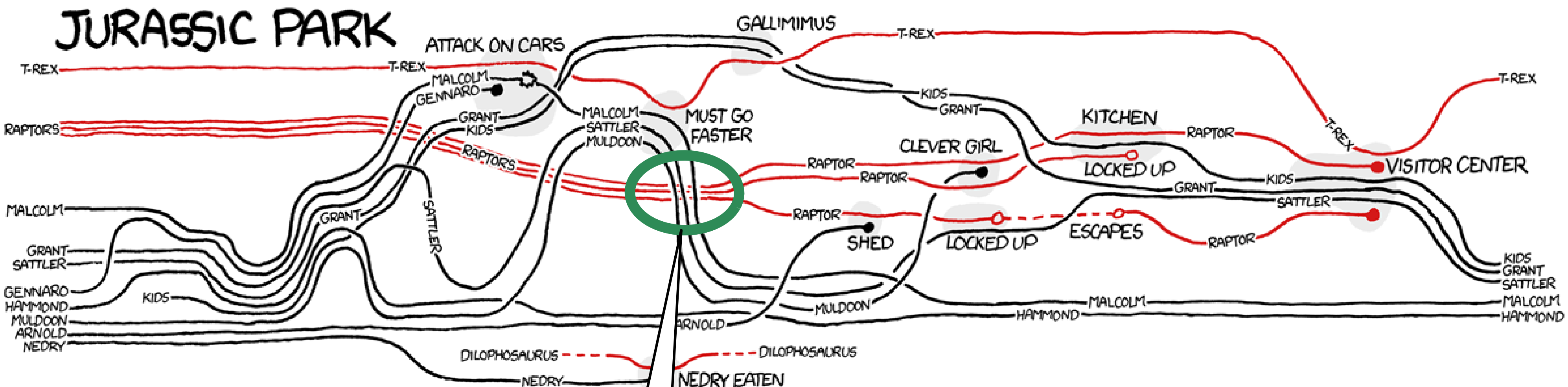

SURI
KOSTITSYNA
POLISHCHUK
SCHULZ
STRASH
NÖLLENBURG
FINK
WOLFF
LIPP
VAN DIJK
FISCHER
MARKFELDER
RAVSKY
GD 2015
SEMINAR GRAPH VIZ
GD REVIEWS
MERGING
YOU ARE HERE



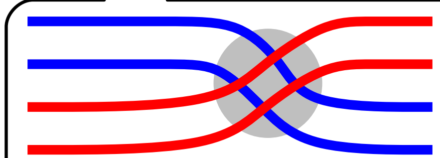
JURASSIC PARK
T-REX
RAPTORS
MALCOLM
GRANT
SATTLER
GENNARO
HAMMOND
MULDOON
ARNOLD
NEDRY
KIDS
ATTACK ON CARS
GALLIMIMUS
MUST GO FASTER
CLEVER GIRL
KITCHEN
LOCKED UP
VISITOR CENTER
SHED
ESCAPES
RAPTOR
DILOPHOSAURUS
NEDRY EATEN


JURASSIC PARK
ATTACK ON CARS
GALLIMIMUS
MUST GO FASTER
CLEVER GIRL
KITCHEN
LOCKED UP
VISITOR CENTER
SHED
LOCKED UP
ESCAPES
NEDRY EATEN
T-REX
RAPTORS
MALCOLM
GRANT
SATTLER
GENNARO
HAMMOND
MULDOON
ARNOLD
NEDRY
KIDS
RAPTOR
DILOPHOSAURUS


JURASSIC PARK
T-REX
RAPTORS
MALCOLM
GRANT
SATTLER
GENNARO
HAMMOND
MULDOON
ARNOLD
NEDRY
KIDS
ATTACK ON CARS
GALLIMIMUS
MUST GO FASTER
CLEVER GIRL
KITCHEN
LOCKED UP
VISITOR CENTER
SHED
ESCAPES
DILOPHOSAURUS
NEDRY EATEN
block crossing



We want to minimize block crossings!



block crossing

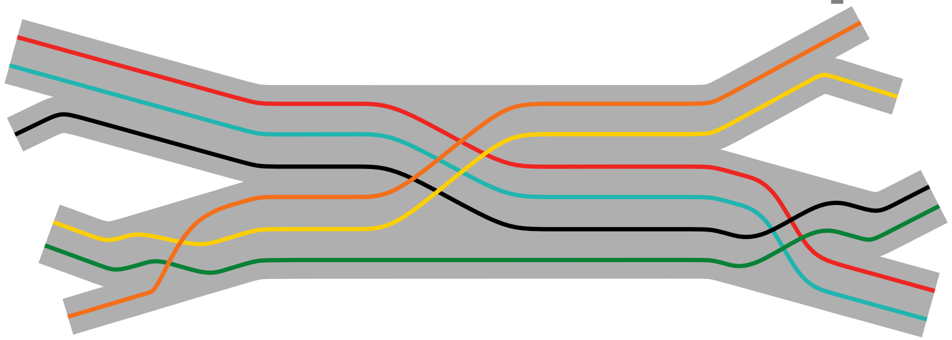
Previous Results – Simple Crossings

[Kostitsyna et al, GD'15]

- NP-hardness
- FPT for #characters
- upper and lower bounds for some cases with pairwise meetings

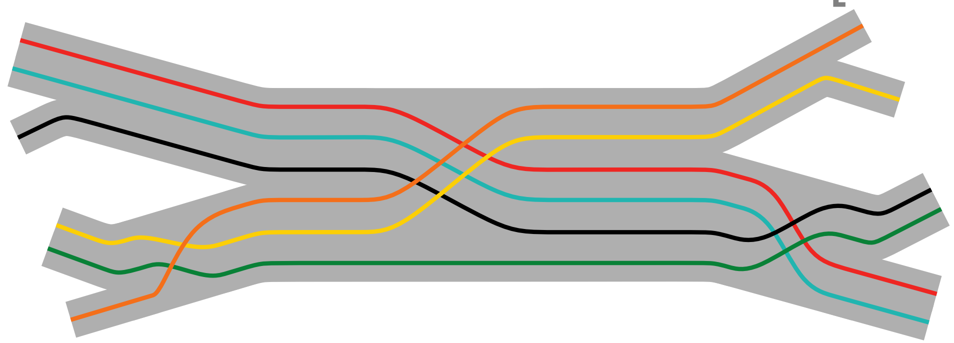
Related Work

- Block crossings for metro lines [Fink, Pupyrev, Wolff; 2015]

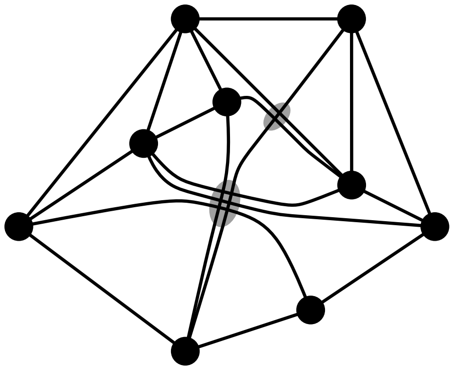


Related Work

- Block crossings for metro lines [Fink, Pupyrev, Wolff; 2015]

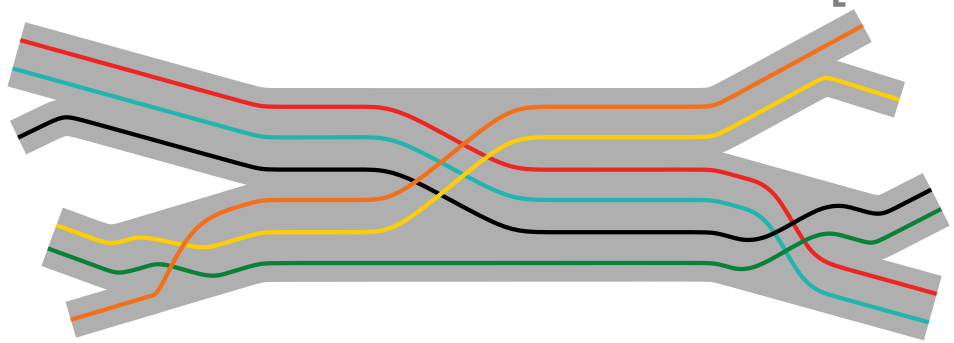


- Bundled Crossings [Fink et al., 2016]

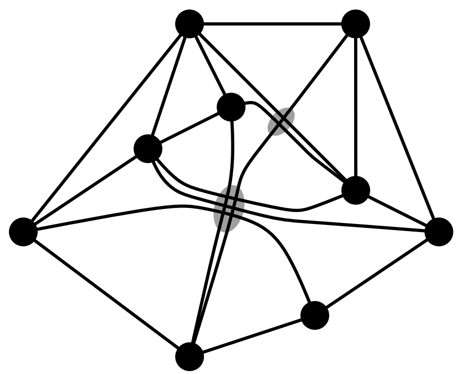


Related Work

- Block crossings for metro lines [Fink, Pupyrev, Wolff; 2015]



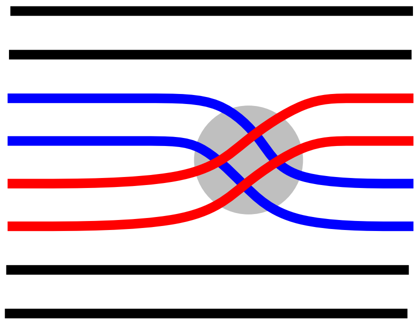
- Bundled Crossings [Fink et al., 2016]



- Bundled Crossing Number [Alam, Fink, Pupyrev; next talk]

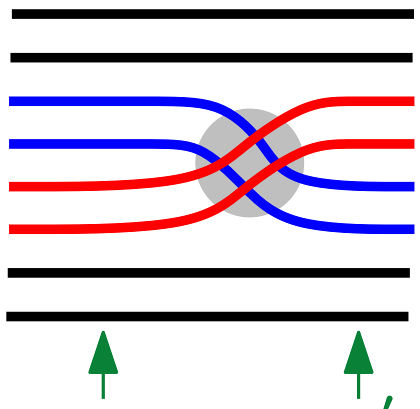
Storylines & Block Crossings

- block crossing



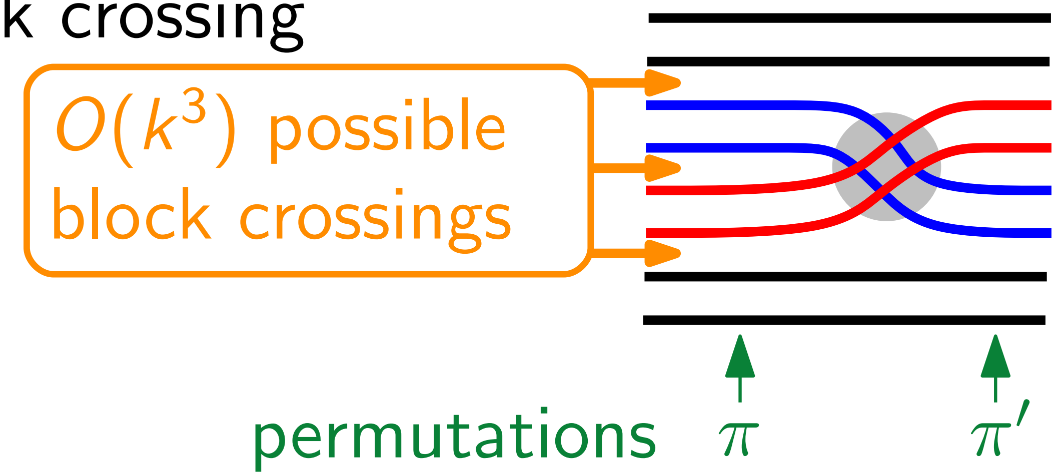
Storylines & Block Crossings

- block crossing



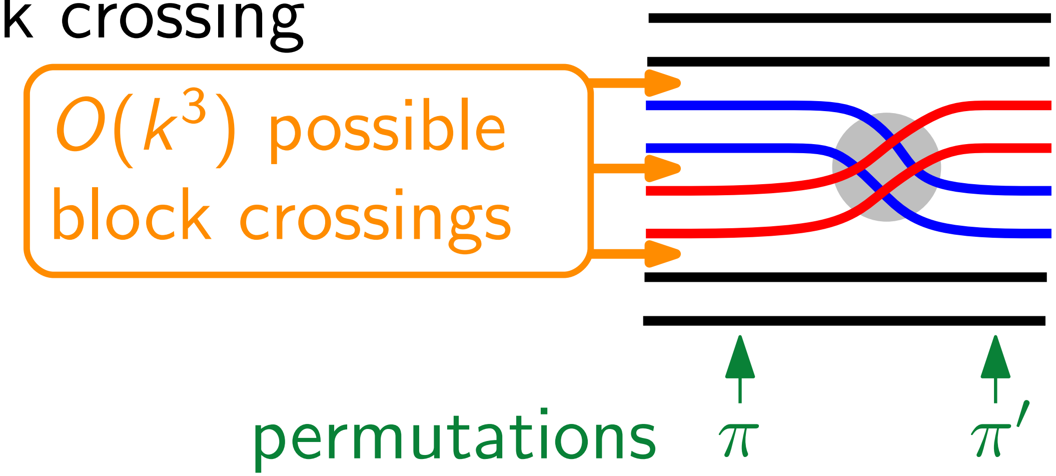
Storylines & Block Crossings

- block crossing

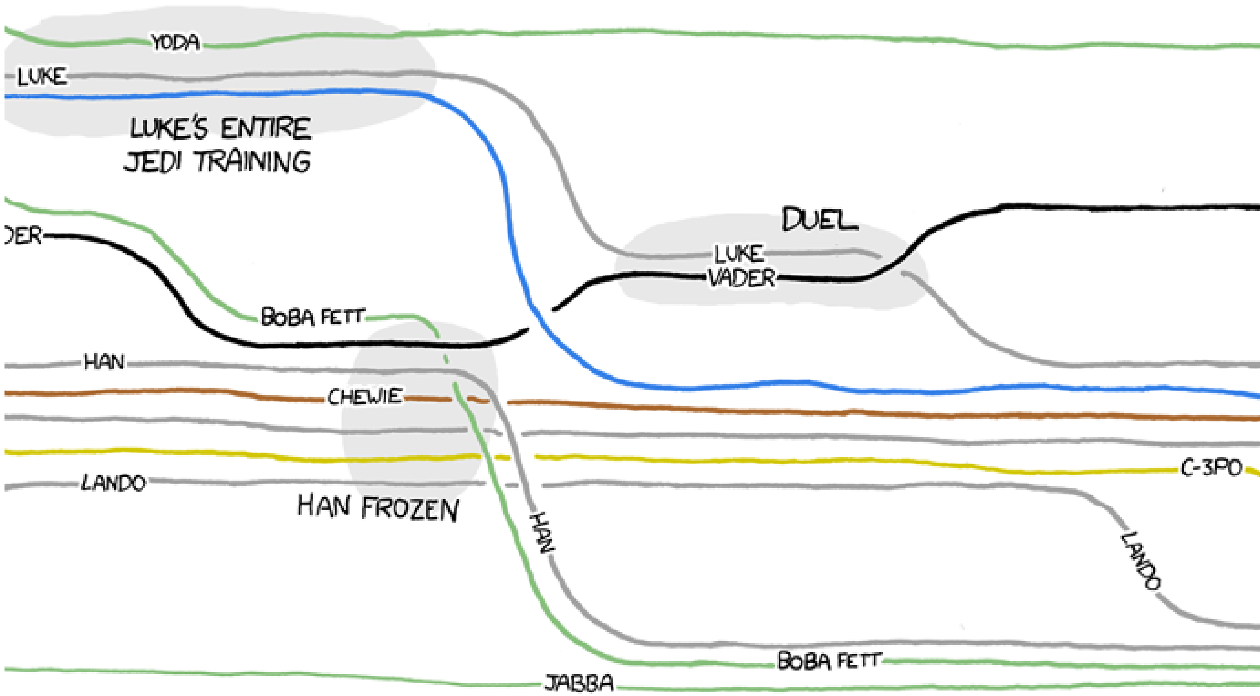


Storylines & Block Crossings

- block crossing

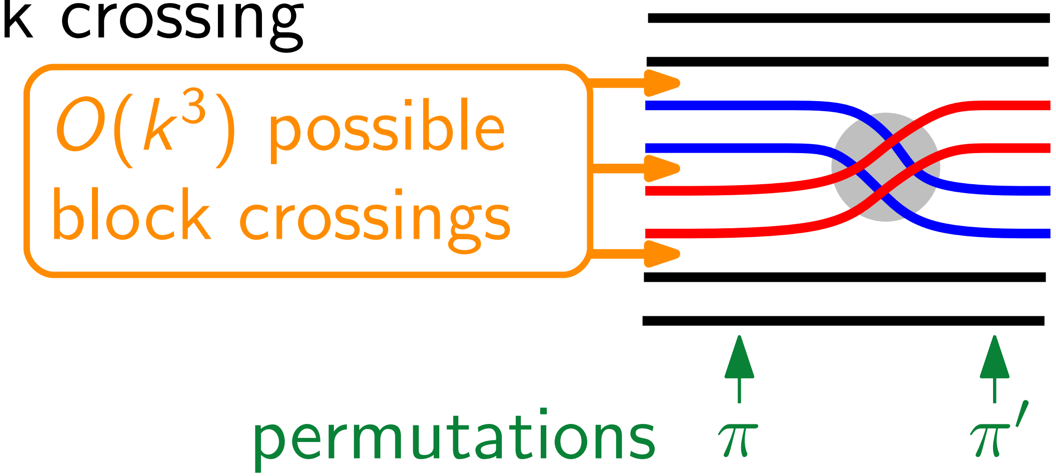


- storyline visualization

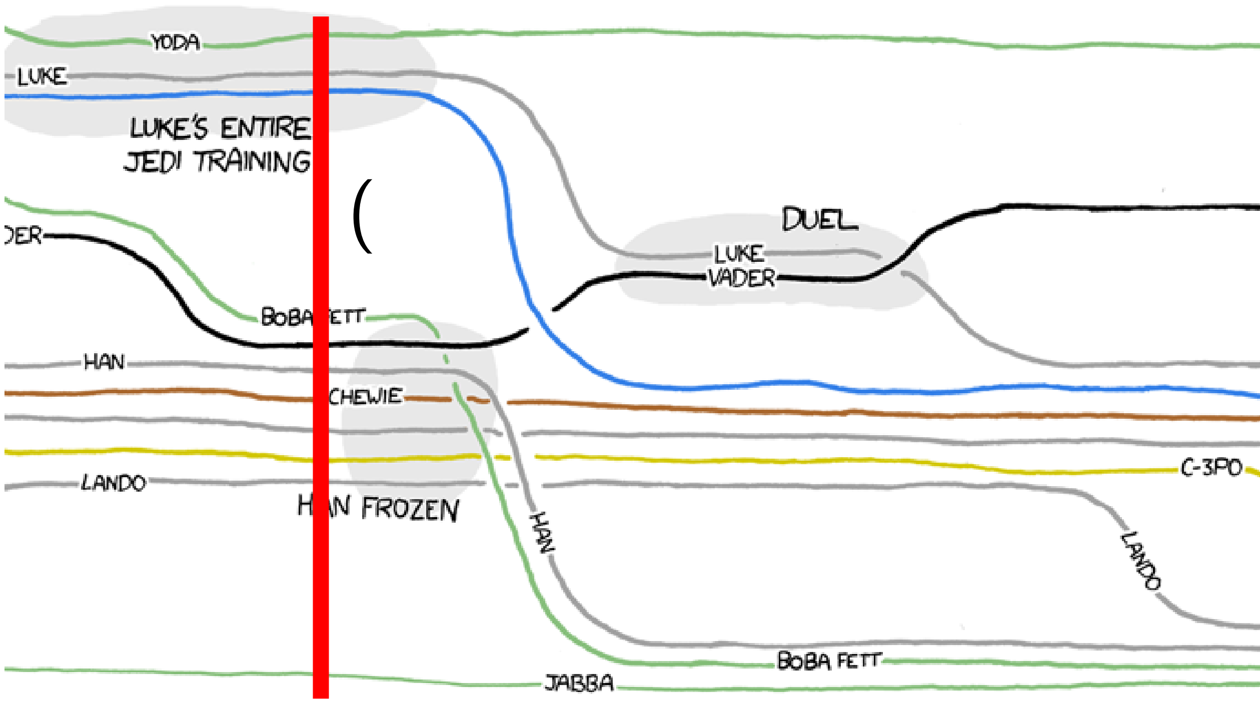


Storylines & Block Crossings

- block crossing



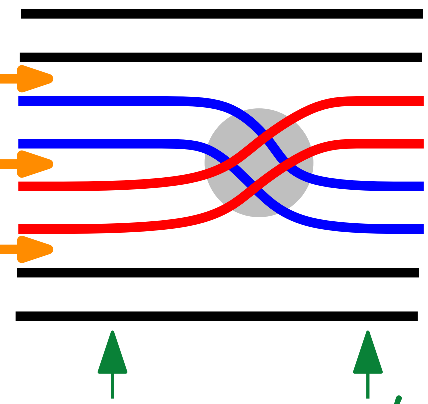
- storyline visualization



Storylines & Block Crossings

- block crossing

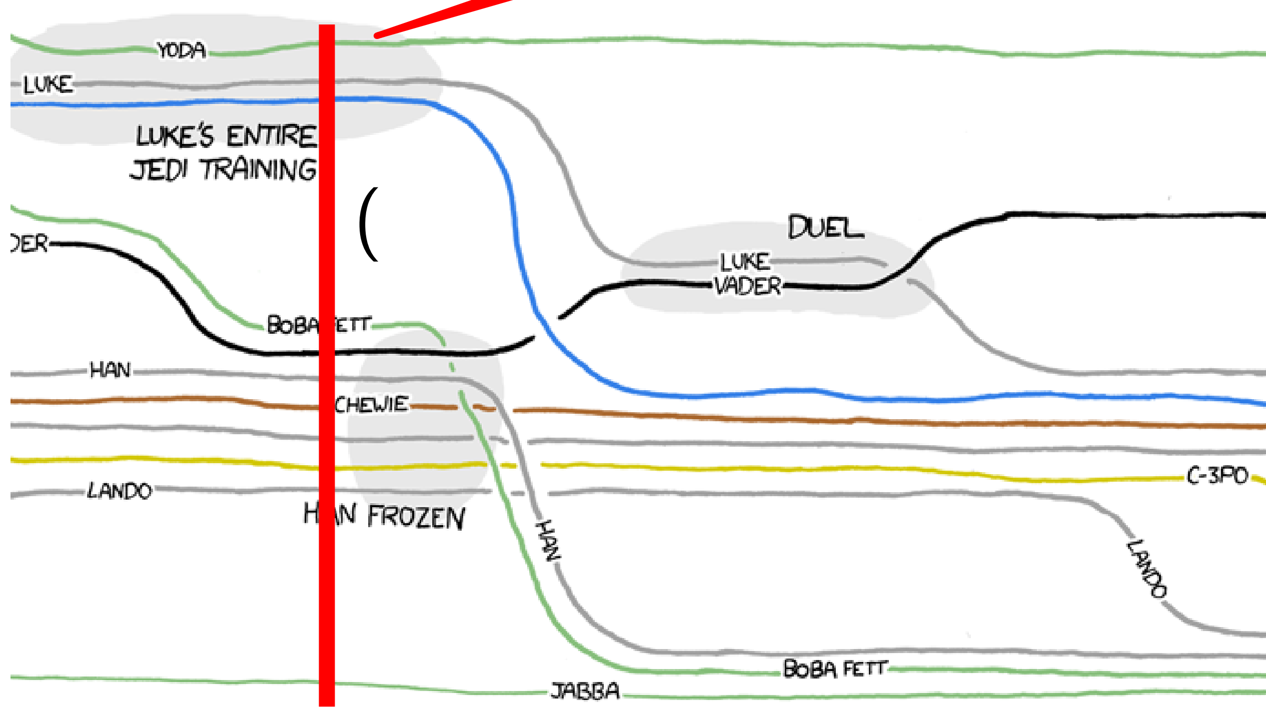
$O(k^3)$ possible block crossings



permutations π π'

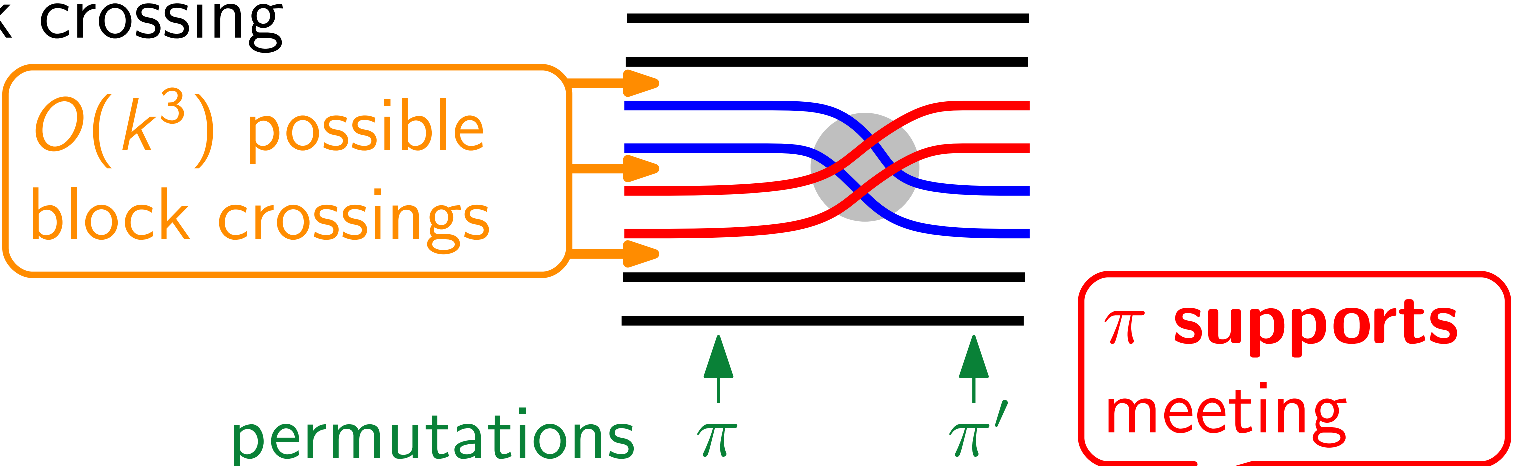
π **supports** meeting

- storyline visualization

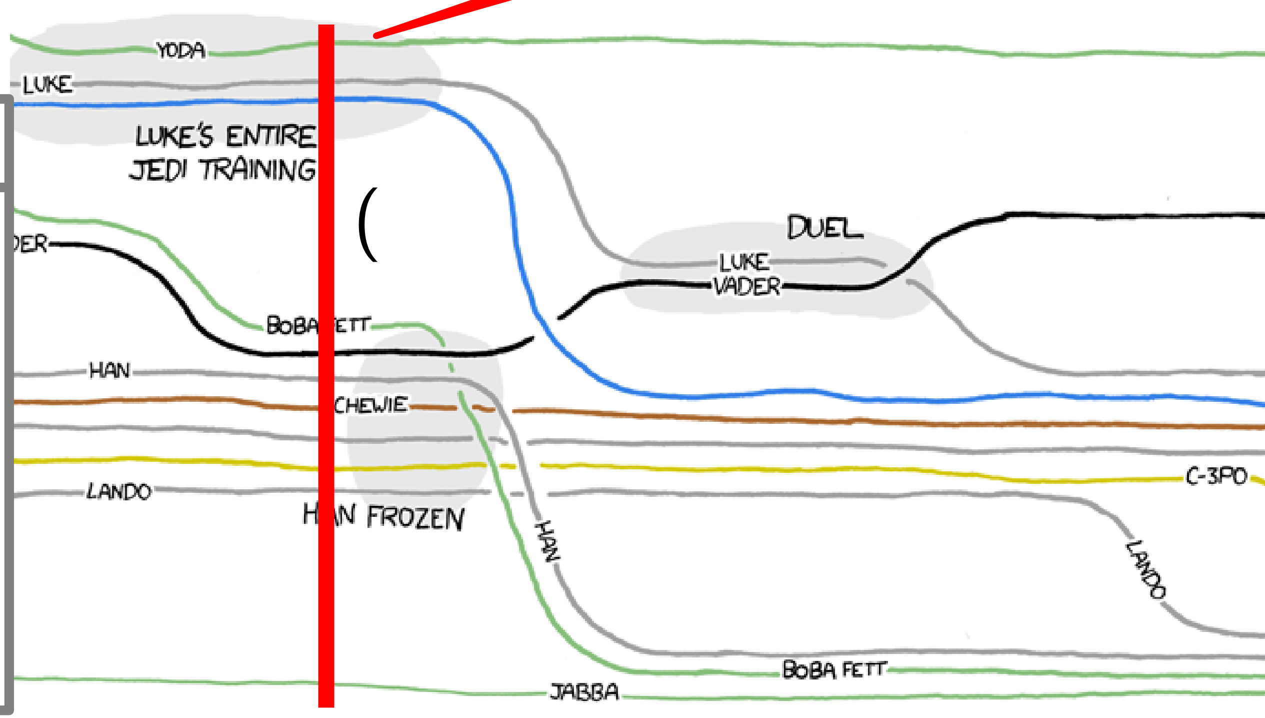


Storylines & Block Crossings

- block crossing



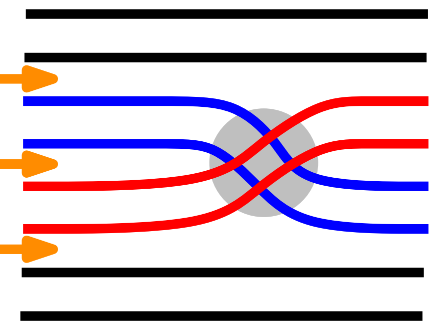
- storyline visualization



Problem definition:

Storylines & Block Crossings

- block crossing

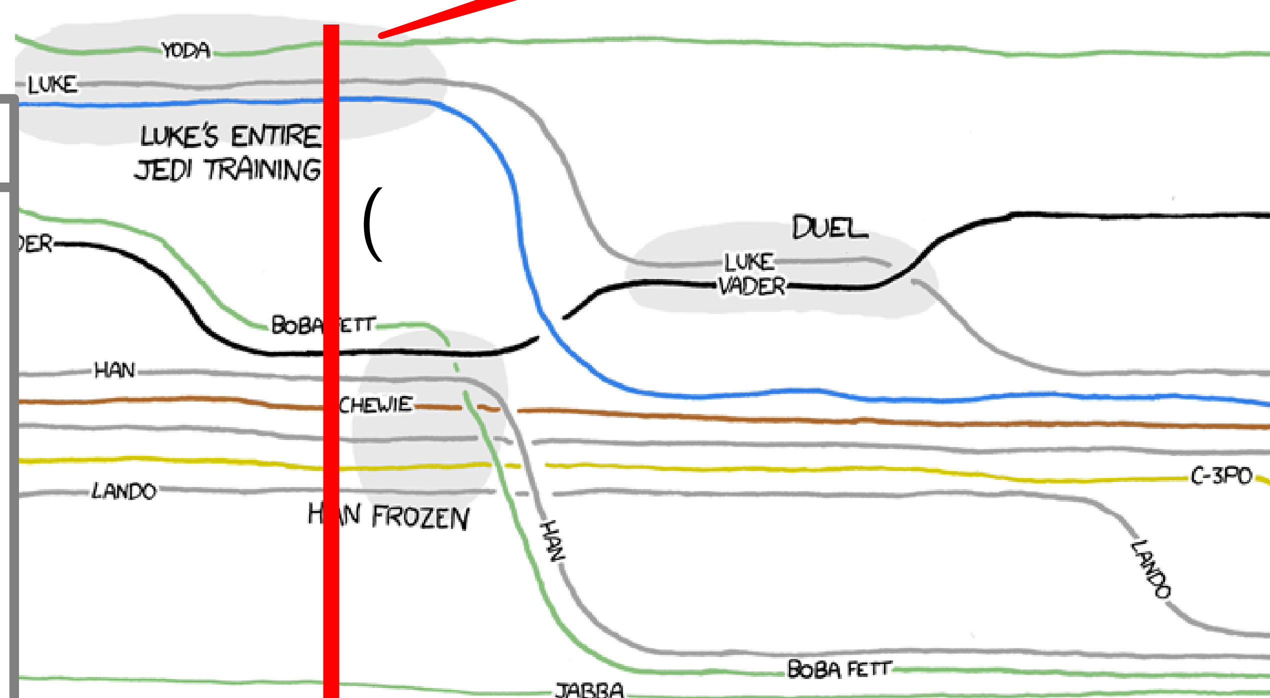


$O(k^3)$ possible block crossings

permutations π π'

- storyline visualization

π **supports** meeting

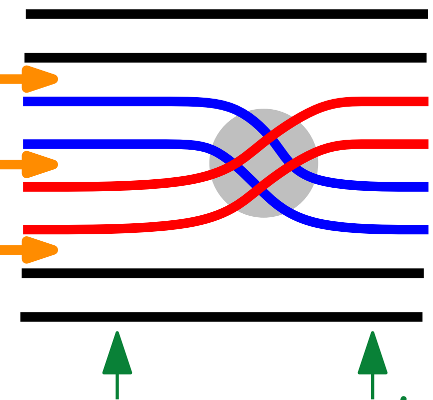


Problem definition:

Given n meetings of k characters, find permutations transformed by min. # block crossings.

Storylines & Block Crossings

- block crossing

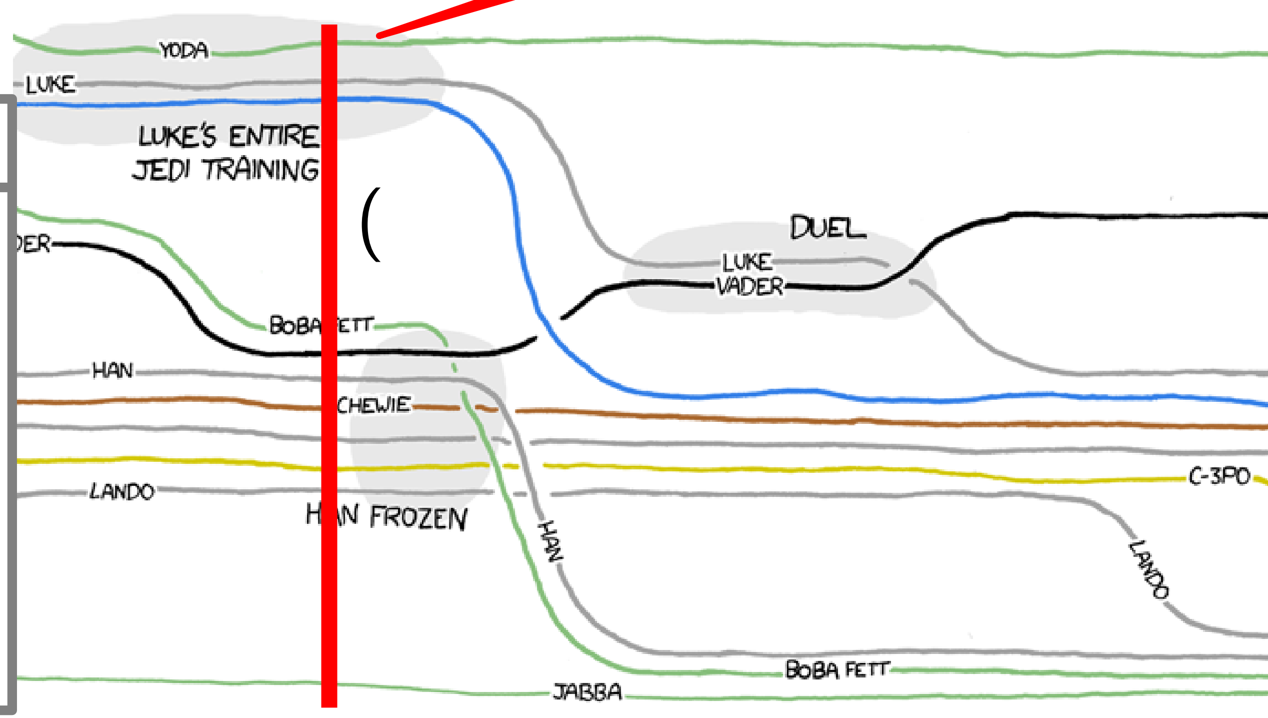


$O(k^3)$ possible block crossings

permutations π π'

π **supports** meeting

- storyline visualization



Problem definition:

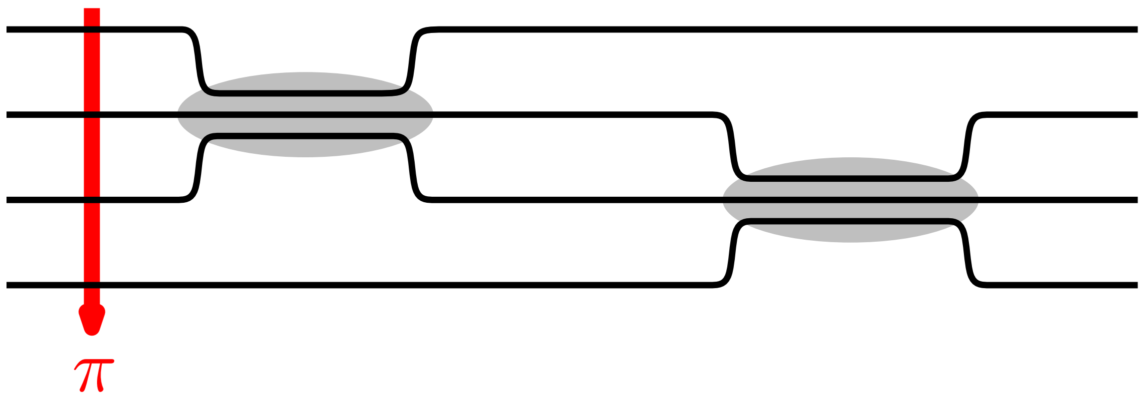
Given n meetings of k characters, find permutations transformed by min. # block crossings.
(Must support all meetings.)

Our Results

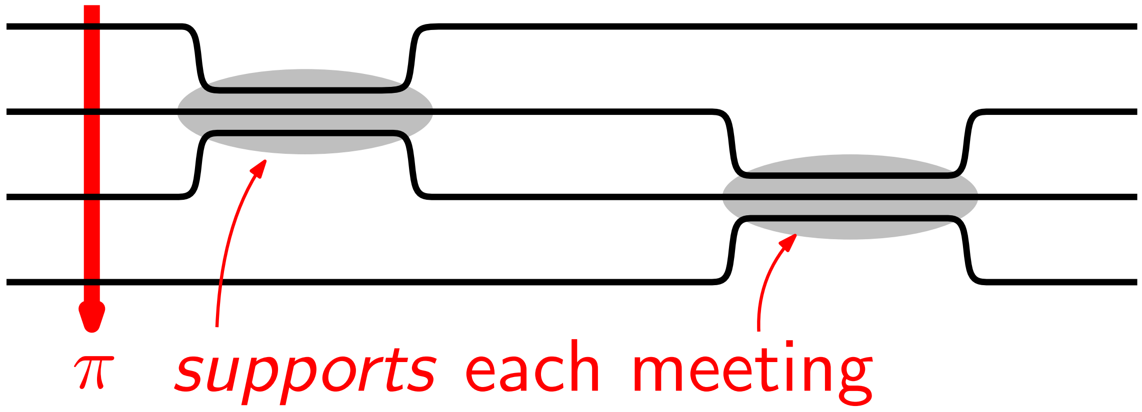
- recognize crossing-free instances
- NP-hardness
- approximation
- FPT/exact algorithms
- greedy heuristic for pairwise meetings

Crossing-Free Storylines Visualizations

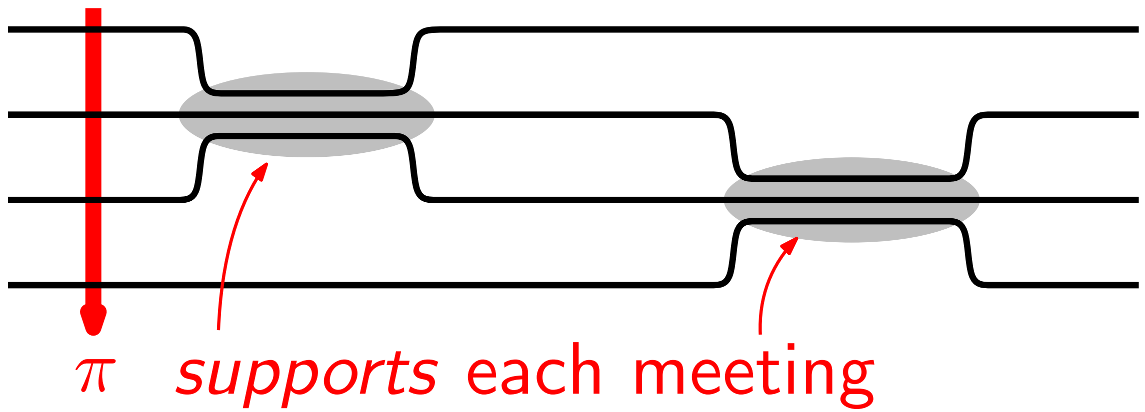
Crossing-Free Storylines Visualizations



Crossing-Free Storylines Visualizations

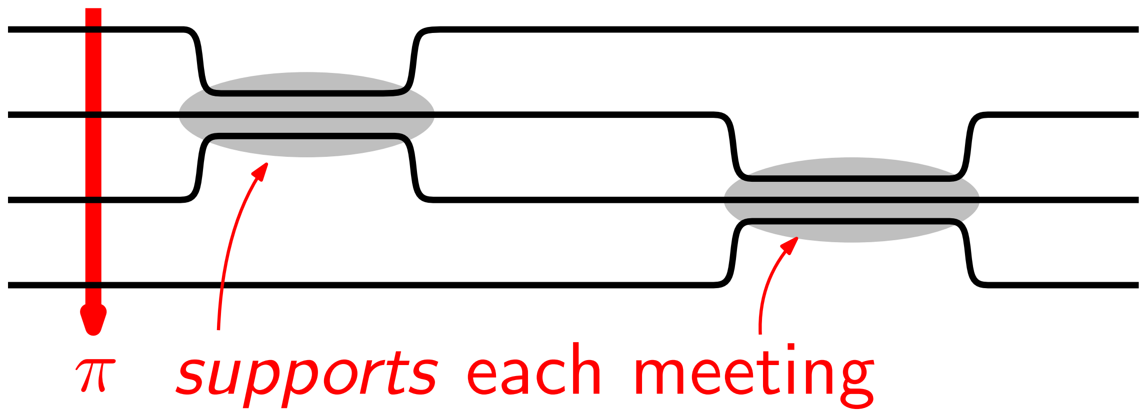


Crossing-Free Storylines Visualizations



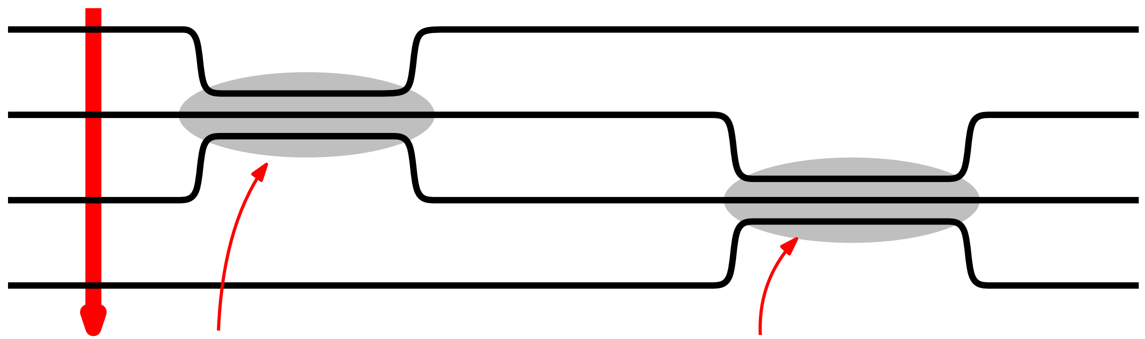
- *group hypergraph* $\mathcal{H} = (C, \Gamma)$ is *interval hypergraph*

Crossing-Free Storylines Visualizations



- *group hypergraph* $\mathcal{H} = (C, \Gamma)$ is *interval hypergraph*

Crossing-Free Storylines Visualizations



- *group hypergraph* $\mathcal{H} = (C, \Gamma)$ is *interval hypergraph*

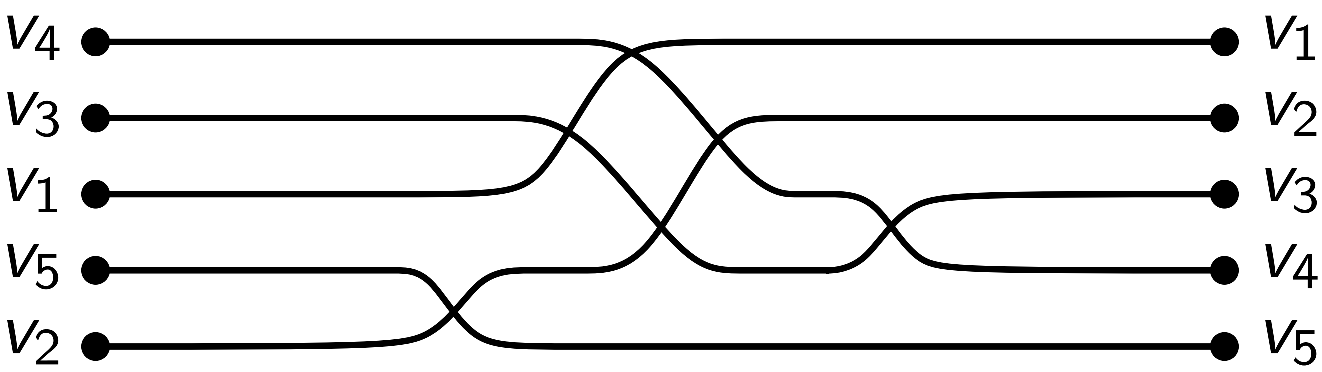
groups that meet

- interval hypergraph property can be checked in $O(k^2)$ time

[Trotter, Moore, 1976]

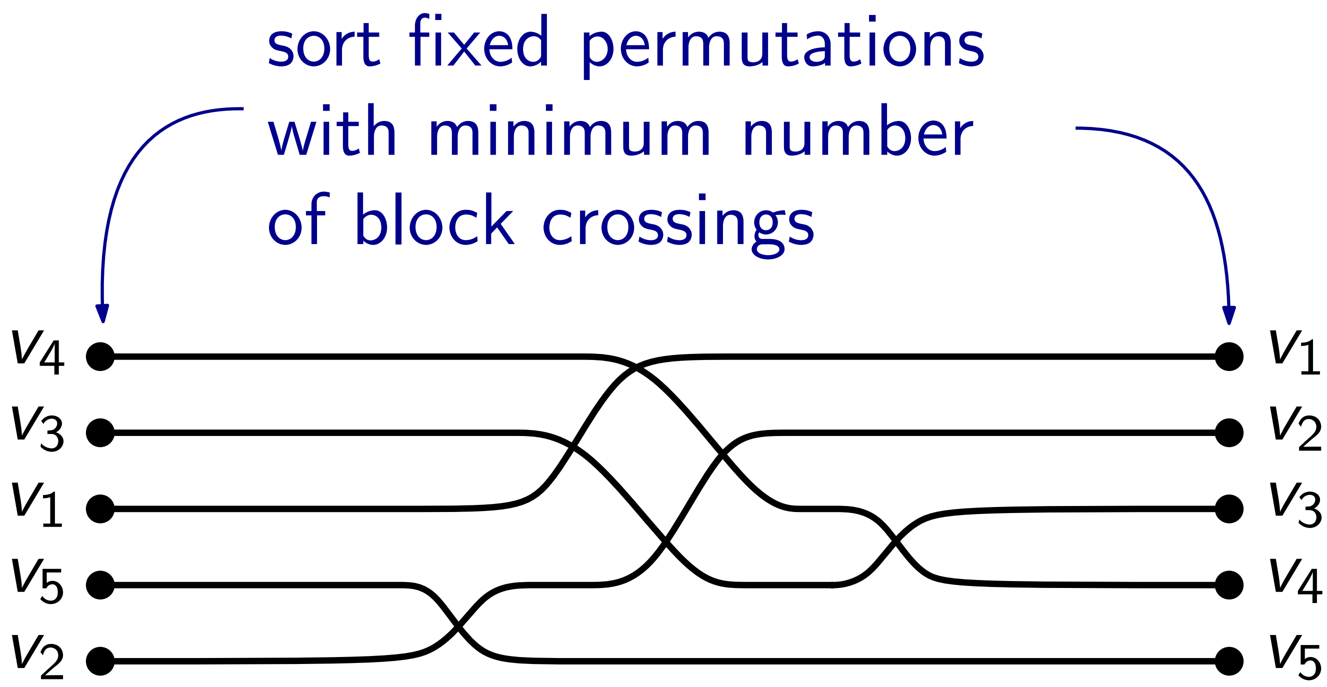
Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*



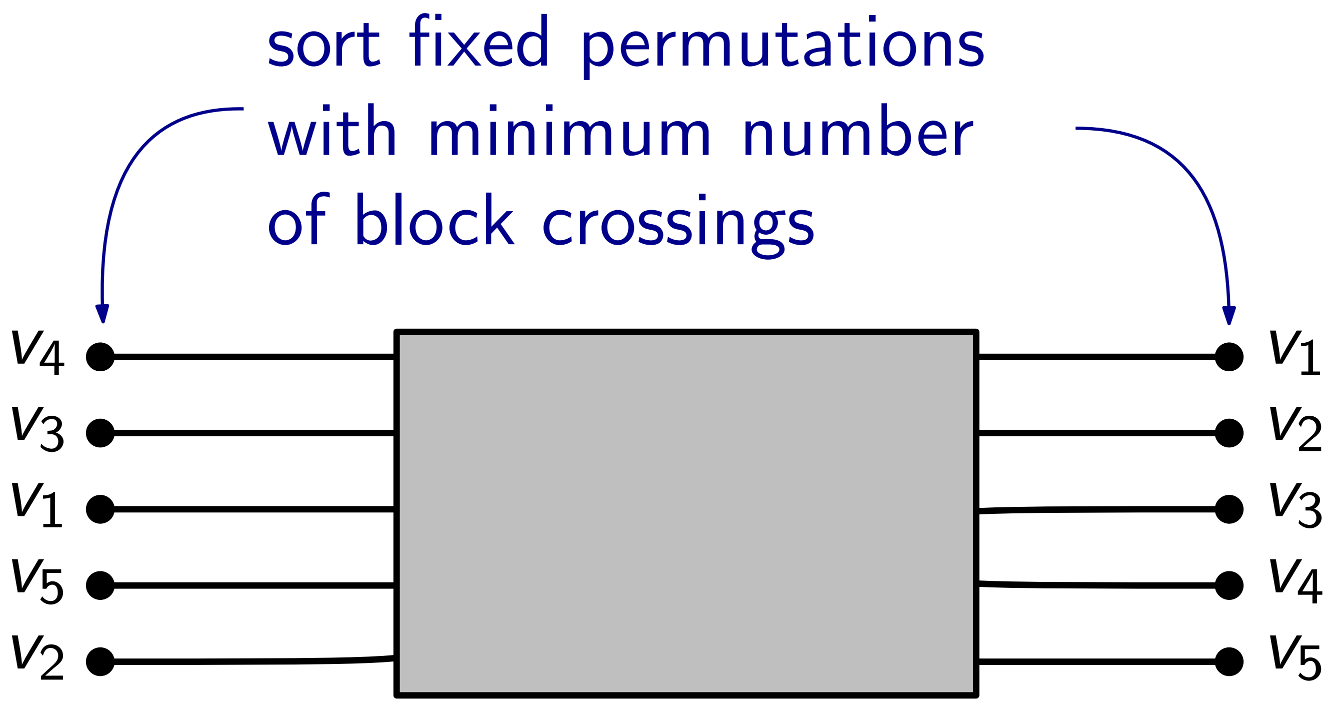
Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*



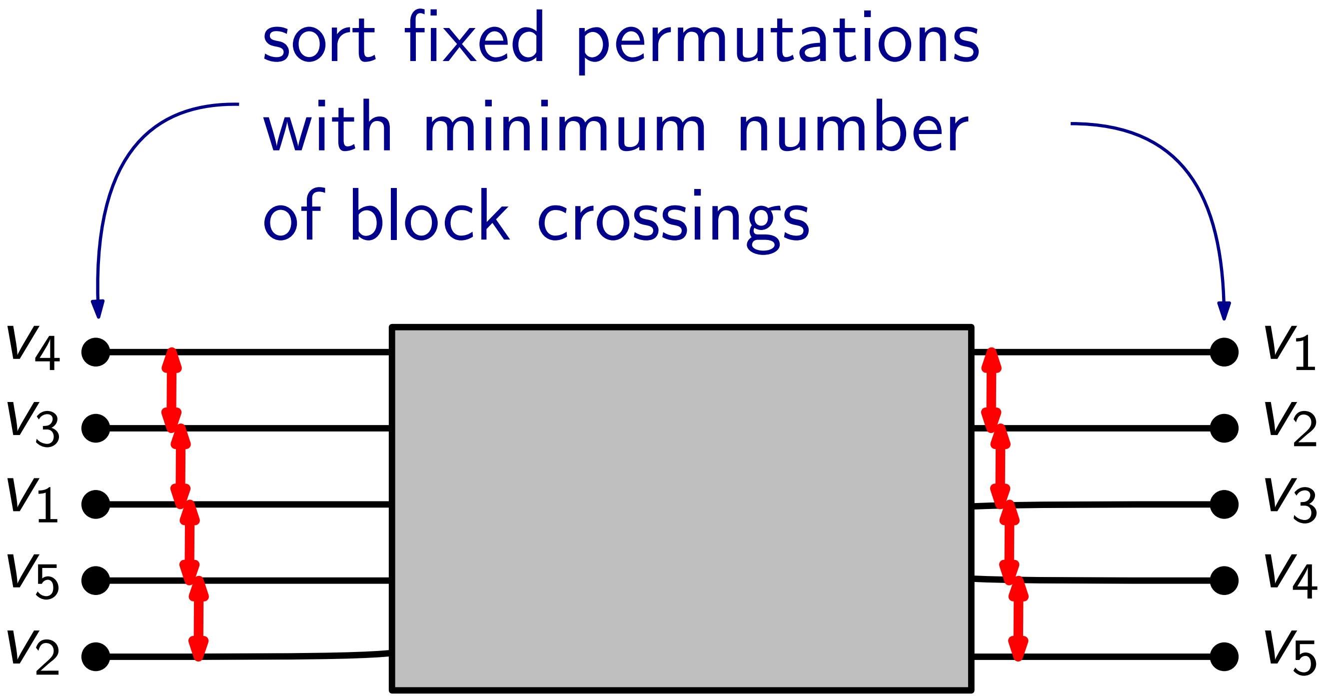
Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*



Minimizing Block Crossings is NP-hard

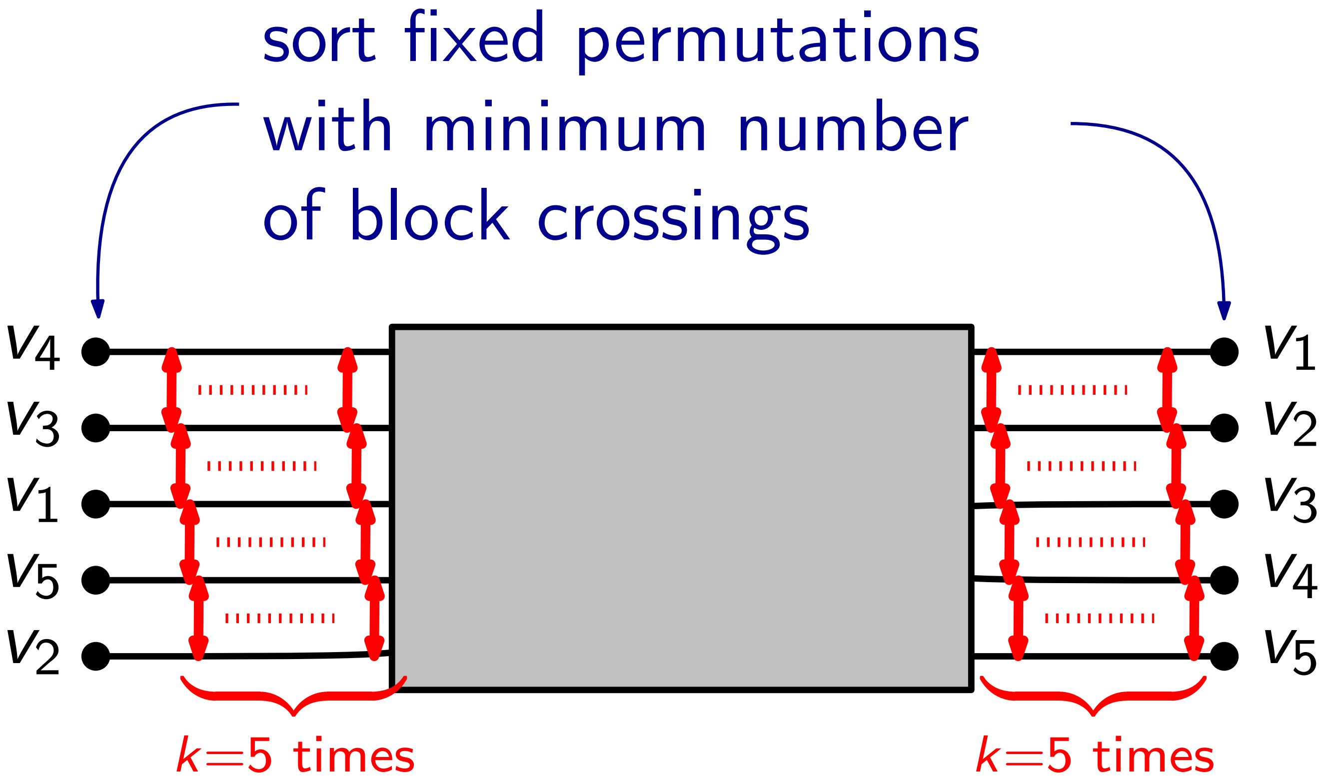
- Reduction from *Sorting by Transpositions*



- fix permutations by repeated meetings

Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*



- fix permutations by repeated meetings

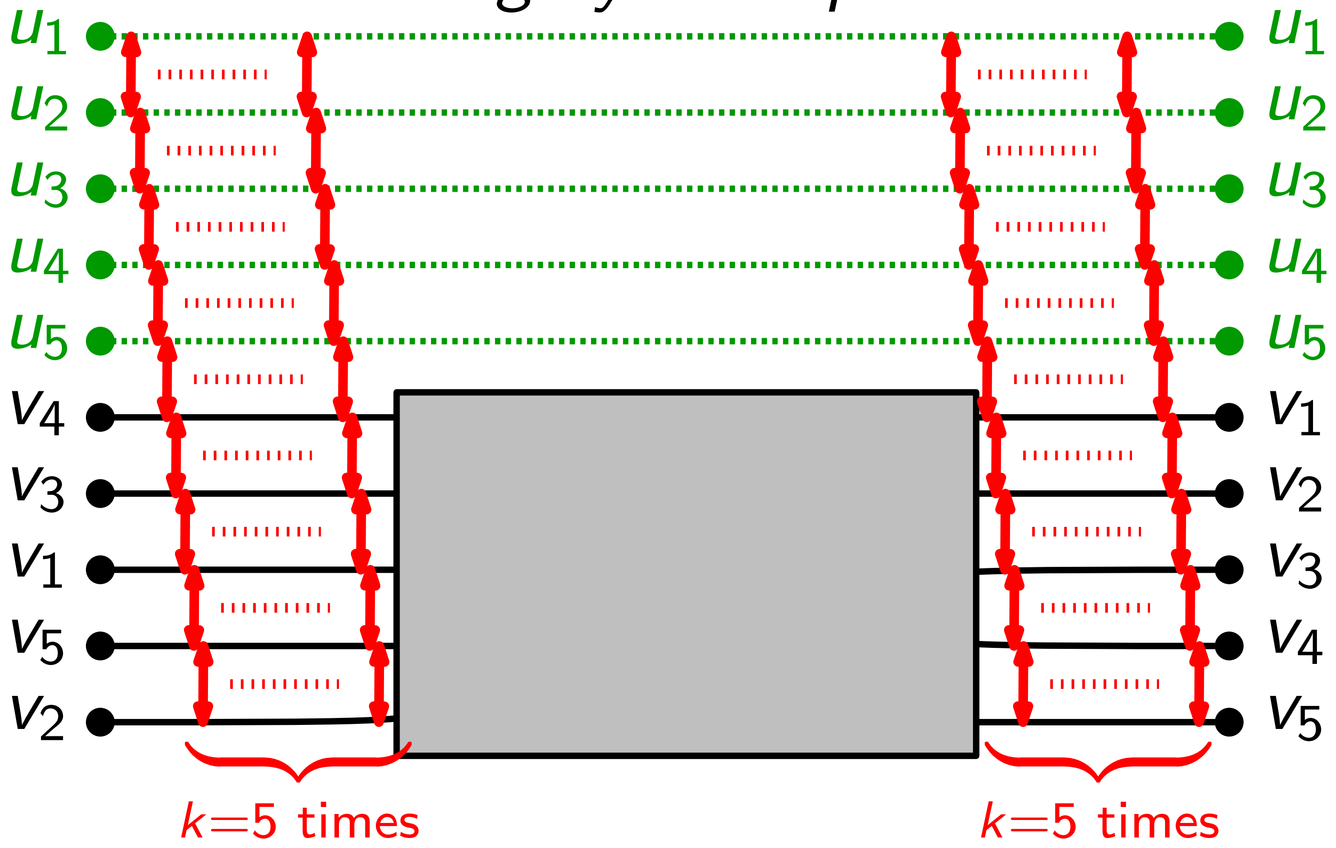
Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*

- fix permutations by repeated meetings
- add frame to prevent reversal

Minimizing Block Crossings is NP-hard

- Reduction from *Sorting by Transpositions*



- fix permutations by repeated meetings
- add frame to prevent reversal

Approximation Algorithm

- all meetings of size $\leq d$ (constant)
- no repeated meetings

Approximation Algorithm

- all meetings of size $\leq d$ (constant)
- no repeated meetings

idea:

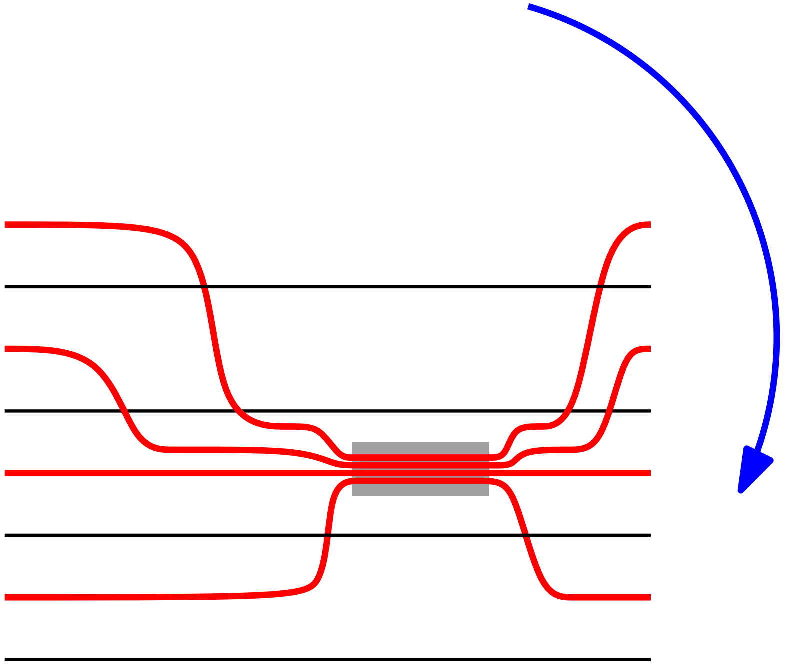
1. choose starting order π that supports many meetings
2. *temporarily* change order for each unsupported meeting

Approximation Algorithm

- all meetings of size $\leq d$ (constant)
- no repeated meetings

idea:

1. choose starting order π that supports many meetings
2. *temporarily* change order for each unsupported meeting



$\leq 2(d-1)$ block crossings

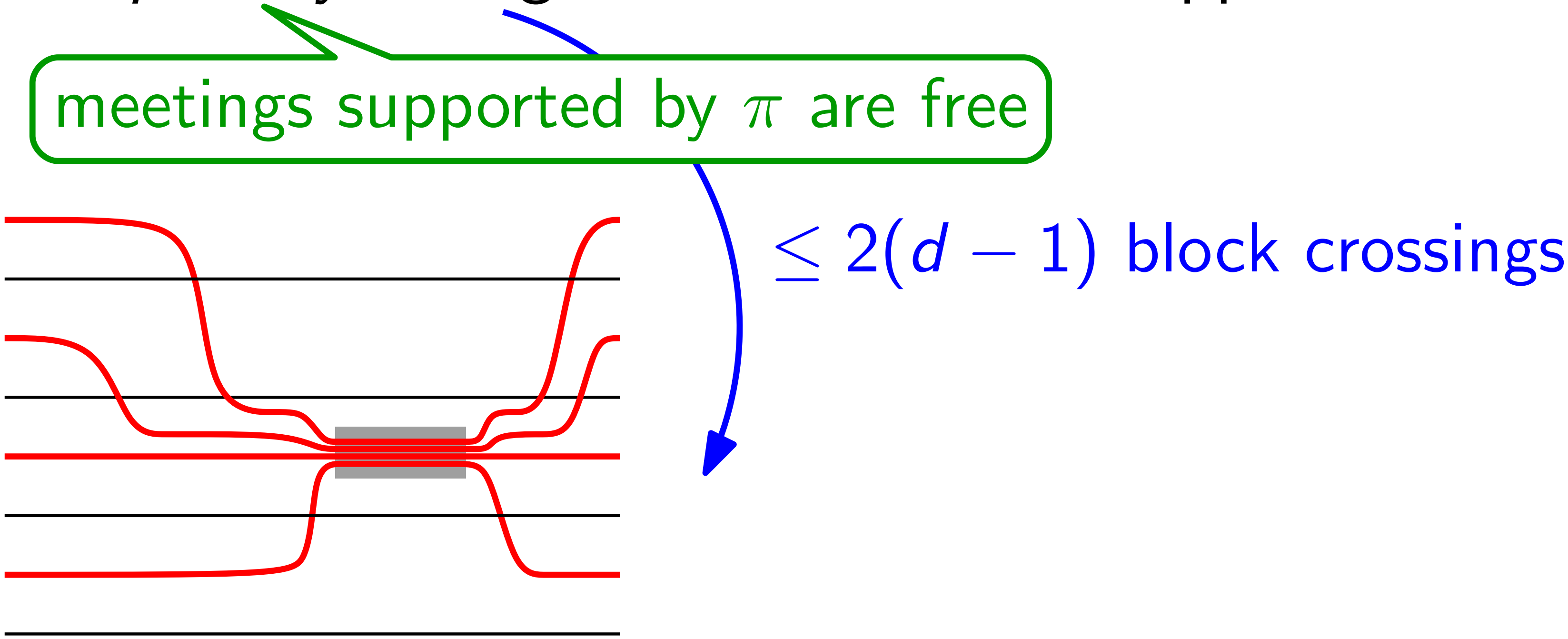
Approximation Algorithm

- all meetings of size $\leq d$ (constant)
- no repeated meetings

idea:

1. choose starting order π that supports many meetings
2. *temporarily* change order for each unsupported meeting

meetings supported by π are free



Approximation Algorithm

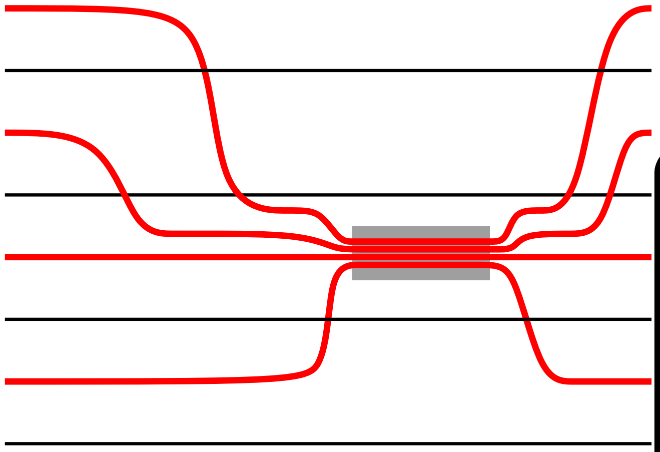
- all meetings of size $\leq d$ (constant)
- no repeated meetings

idea:

1. choose starting order π that supports many meetings
2. *temporarily* change order for each unsupported meeting

meetings supported by π are free

$\leq 2(d-1)$ block crossings



Lemma: starting order π has α unsupported meetings $\Rightarrow$ at least $4\alpha/(3d^2)$ block crossings necessary

Approximation Algorithm

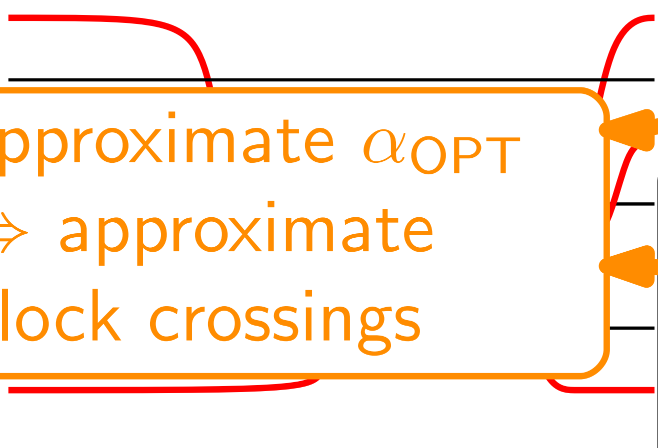
- all meetings of size $\leq d$ (constant)
- no repeated meetings

idea:

1. choose starting order π that supports many meetings
2. *temporarily* change order for each unsupported meeting

meetings supported by π are free

$\leq 2(d-1)$ block crossings



approximate α_{OPT} $\Rightarrow$ approximate block crossings

Lemma: starting order π has α unsupported meetings $\Rightarrow$ at least $4\alpha/(3d^2)$ block crossings necessary

Approximation Algorithm

find π minimizing #unsupported meetings

Approximation Algorithm

find π minimizing #unsupported meetings

$\leftrightarrow$ remove minimum #meetings so that storyline crossing-free

Approximation Algorithm

find π minimizing #unsupported meetings

$\leftrightarrow$ remove minimum #meetings so that storyline crossing-free

$\leftrightarrow$ remove minimum #hyperedges so that $\mathcal{H}$ is interval hypergraph

Approximation Algorithm

find π minimizing #unsupported meetings

$\leftrightarrow$ remove minimum #meetings so that storyline crossing-free

$\leftrightarrow$ remove minimum #hyperedges so that $\mathcal{H}$ is interval hypergraph

Theorem: Interval Hypergraph Edge Deletion admits a $(d+1)$-approximation (constant rank d).

Approximation Algorithm

find π minimizing #unsupported meetings

$\leftrightarrow$ remove minimum #meetings so that storyline crossing-free

$\leftrightarrow$ remove minimum #hyperedges so that $\mathcal{H}$ is interval hypergraph

Theorem: Interval Hypergraph Edge Deletion admits a $(d+1)$-approximation (constant rank d).

Theorem: We can find a $(3(d^2-1)d^2/2)$-approximation for the minimum number of block crossings in storyline visualizations in $O(kn)$ time.

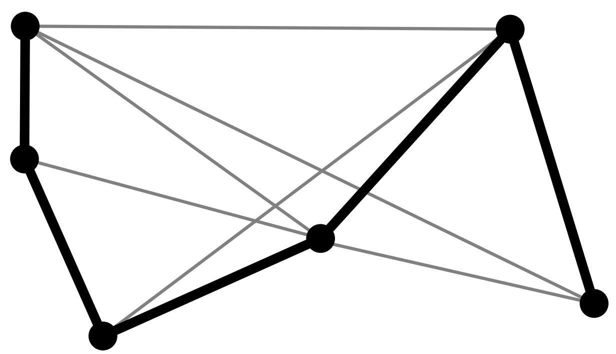
Interval Hypergraph Edge Deletion

- Remove minimum number of hyperedges so that $\mathcal{H} = (V, E)$ becomes interval hypergraph

Interval Hypergraph Edge Deletion

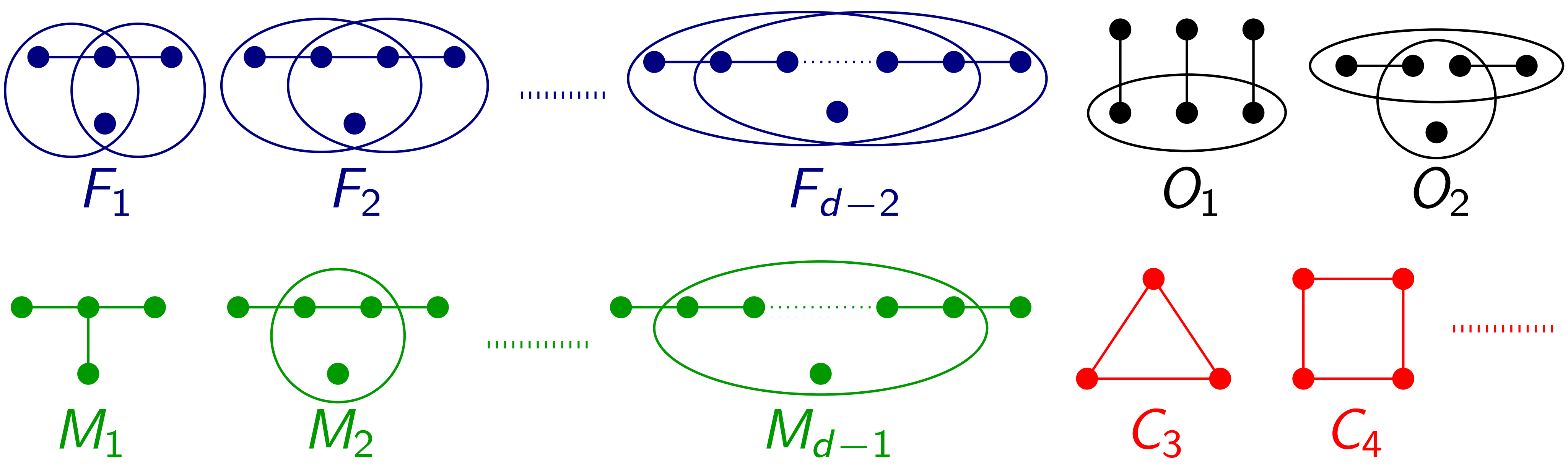
- Remove minimum number of hyperedges so that $\mathcal{H} = (V, E)$ becomes interval hypergraph

NP-hard for graphs:
remove all but $n-1$ edges $\rightarrow$
Hamiltonian path



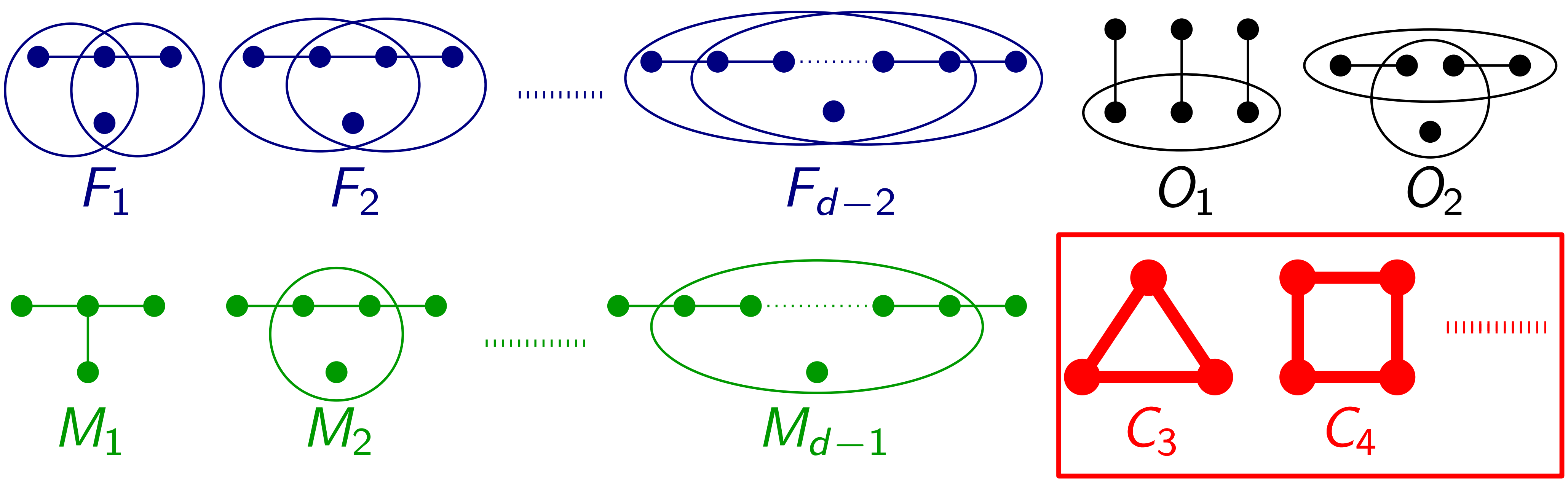
Interval Hypergraph Edge Deletion

- Remove minimum number of hyperedges so that $\mathcal{H} = (V, E)$ becomes interval hypergraph
- characterization of interval hypergraphs by forbidden subhypergraphs



Interval Hypergraph Edge Deletion

- Remove minimum number of hyperedges so that $\mathcal{H} = (V, E)$ becomes interval hypergraph
- characterization of interval hypergraphs by forbidden subhypergraphs

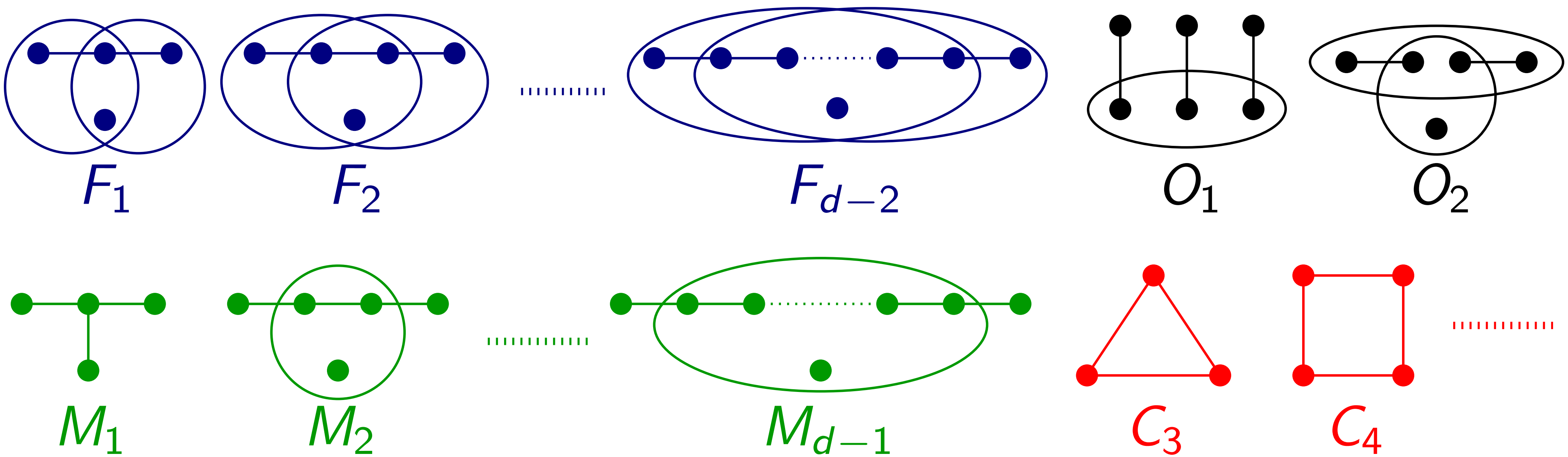


Interval Hypergraph Edge Deletion

- Rem
  $\mathcal{H} =$
- char ... len
  subh

outline:
– iteratively: search for forbidden subhypergraphs except $C_{d+2}, \ldots$ & completely remove them

– result: cyclic generalization of interval hypergraph; break optimally



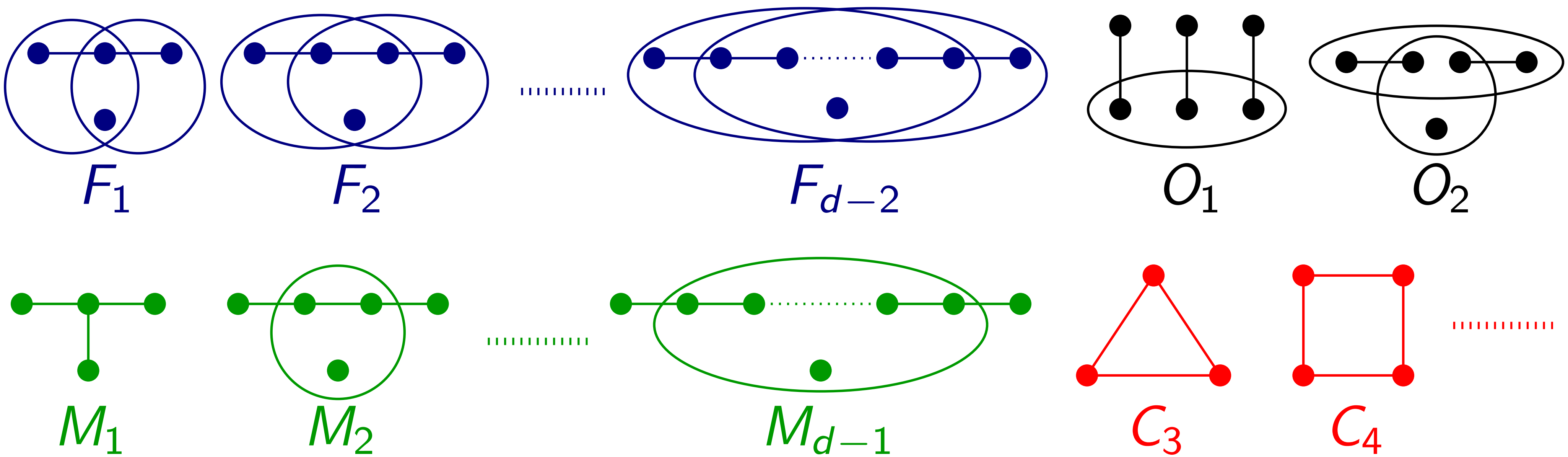
Interval Hypergraph Edge Deletion

- Rem
  $\mathcal{H} =$
- char ... en
  subh

outline:
– iteratively: search for forbidden subhypergraphs except $C_{d+2}, \ldots$ & completely remove them

– result: cyclic generalization of interval hypergraph; break optimally

$\leq d+1$ hyperedges



Interval Hypergraph Edge Deletion

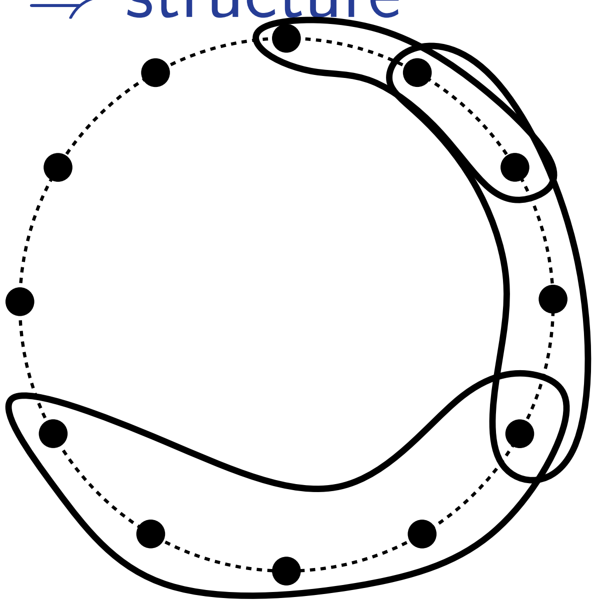
- Rem
  $\mathcal{H} =$
- char ... en
  subh

outline:
– iteratively: search for forbidden subhypergraphs except $C_{d+2}, \ldots$ & completely remove them

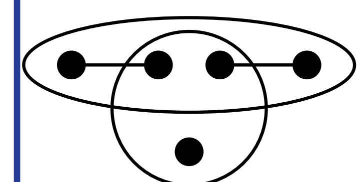
$\leq d+1$ hyperedges

– result: cyclic generalization of interval hypergraph; break optimally

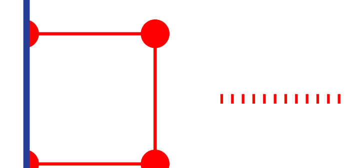
Theorem: no forbidden subhypergraph except $C_{d+2}, \ldots \Rightarrow$ structure



proof skipped (several lemmas & case distinctions)



O_2



C_4

Interval Hypergraph Edge Deletion

- Rem
  $\mathcal{H} =$
- char

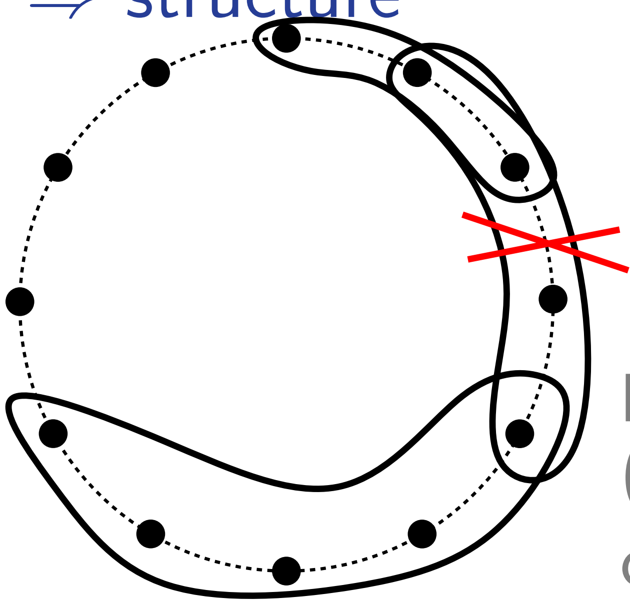
outline:
– iteratively: search for forbidden subhypergraphs except $C_{d+2}, \dots$ &

$\leq d + 1$ hyperedges

hypergraph; break optimally

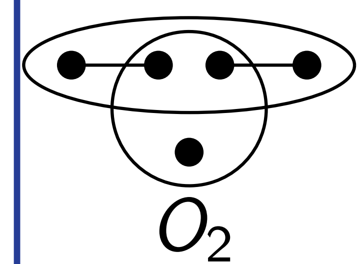
Theorem: Interval Hypergraph Edge Deletion admits a $(d+1)$-approximation (constant rank d).

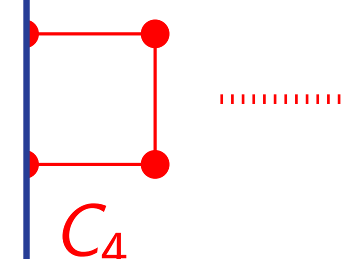
Theorem: no forbidden subhypergraph except $C_{d+2}, \dots \Rightarrow$ structure

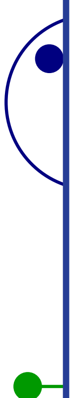


can cut optimally

proof skipped (several lemmas & case distinctions)





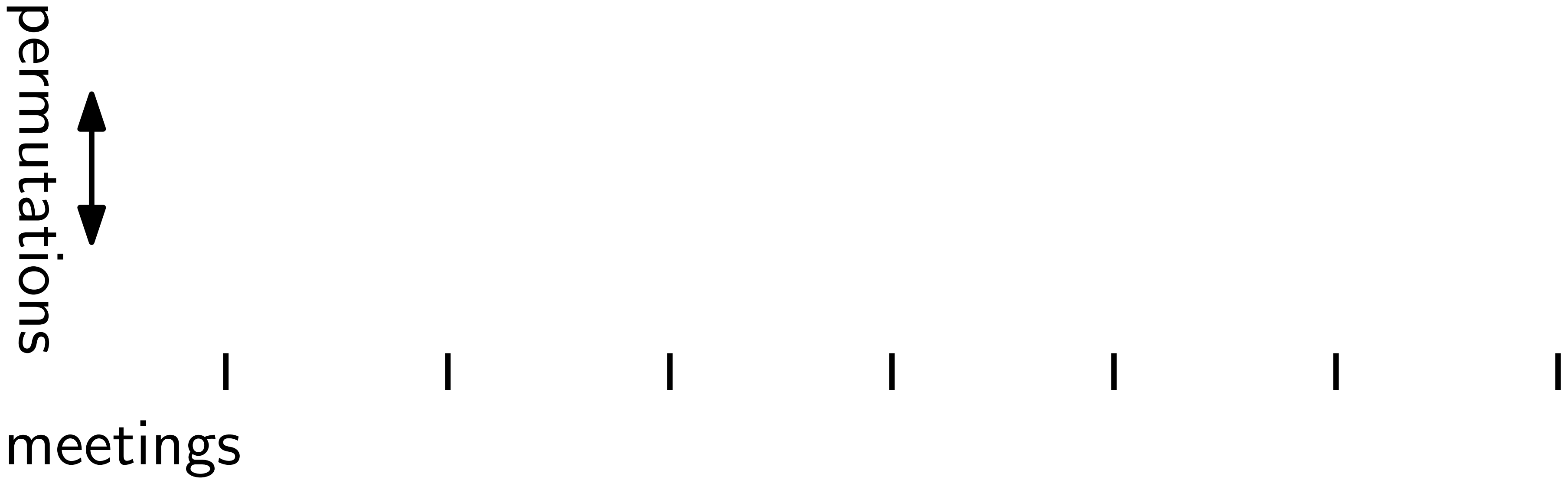


Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings

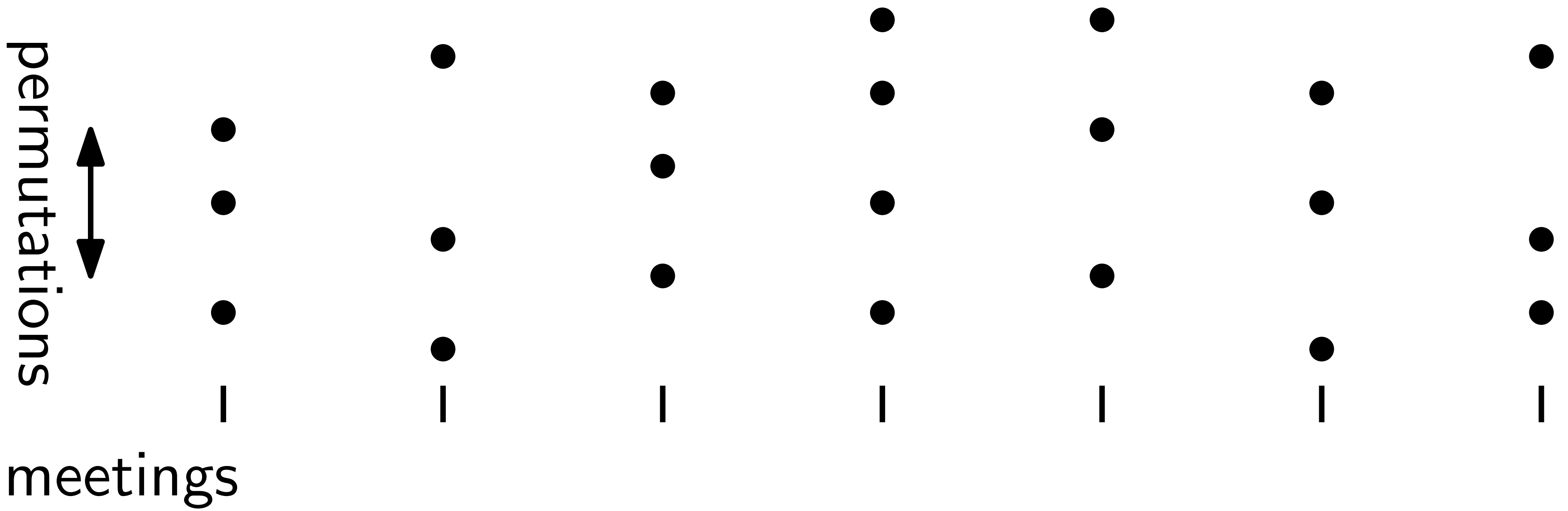
Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



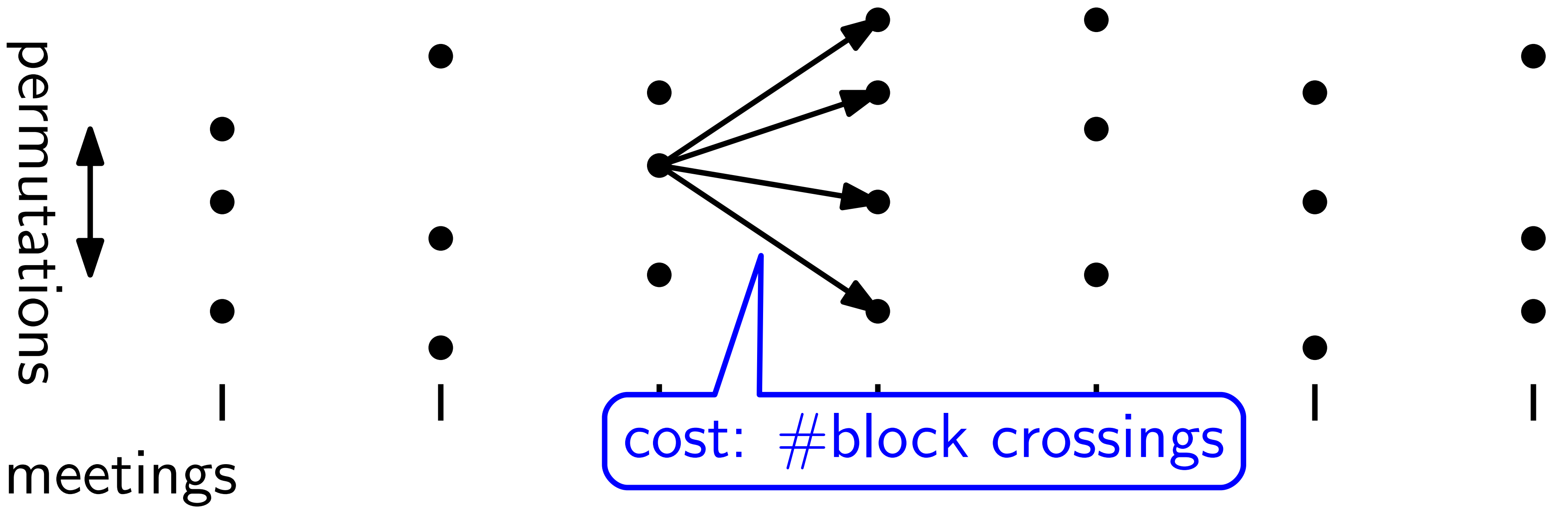
Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



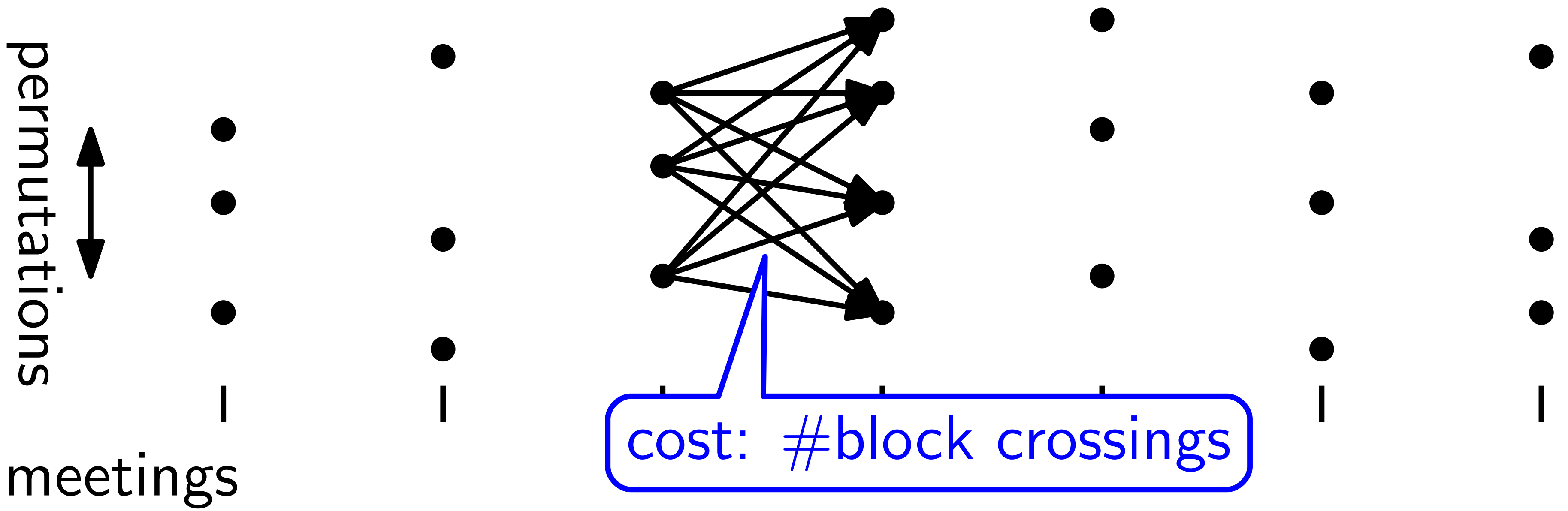
Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



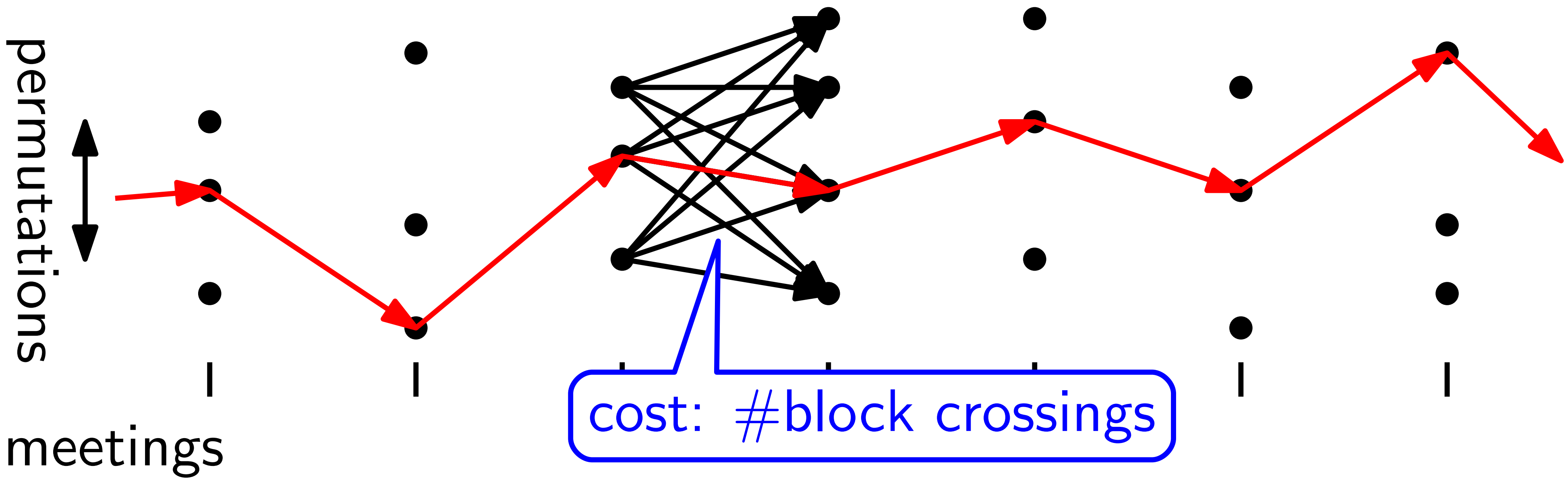
Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



Exact Algorithm

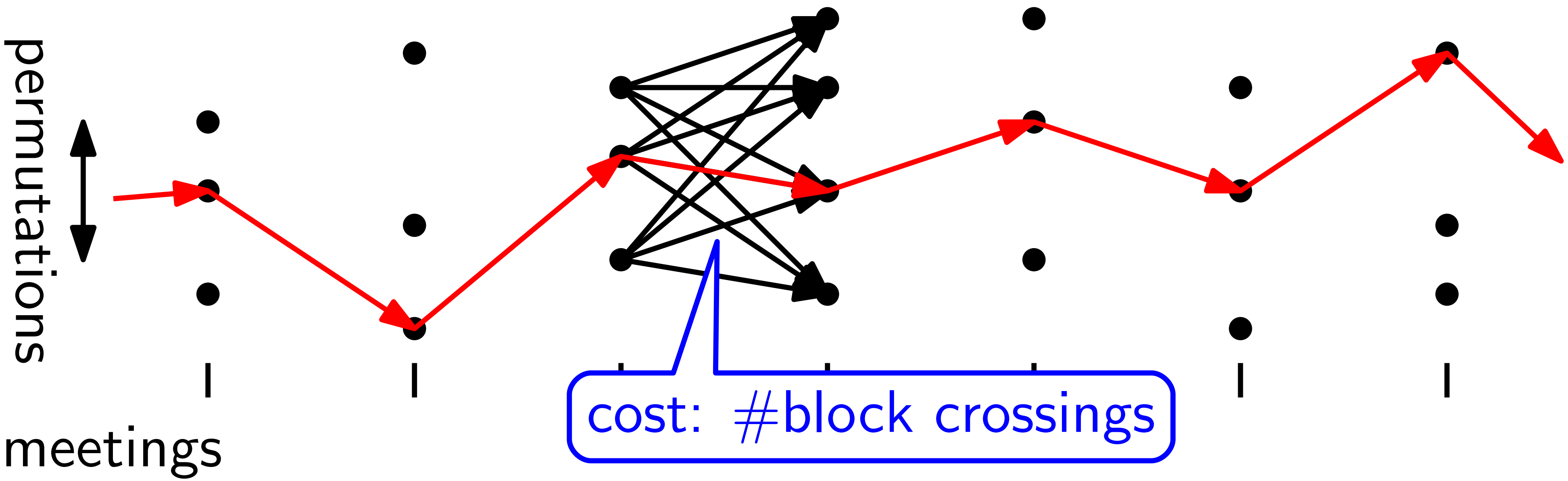
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path

Exact Algorithm

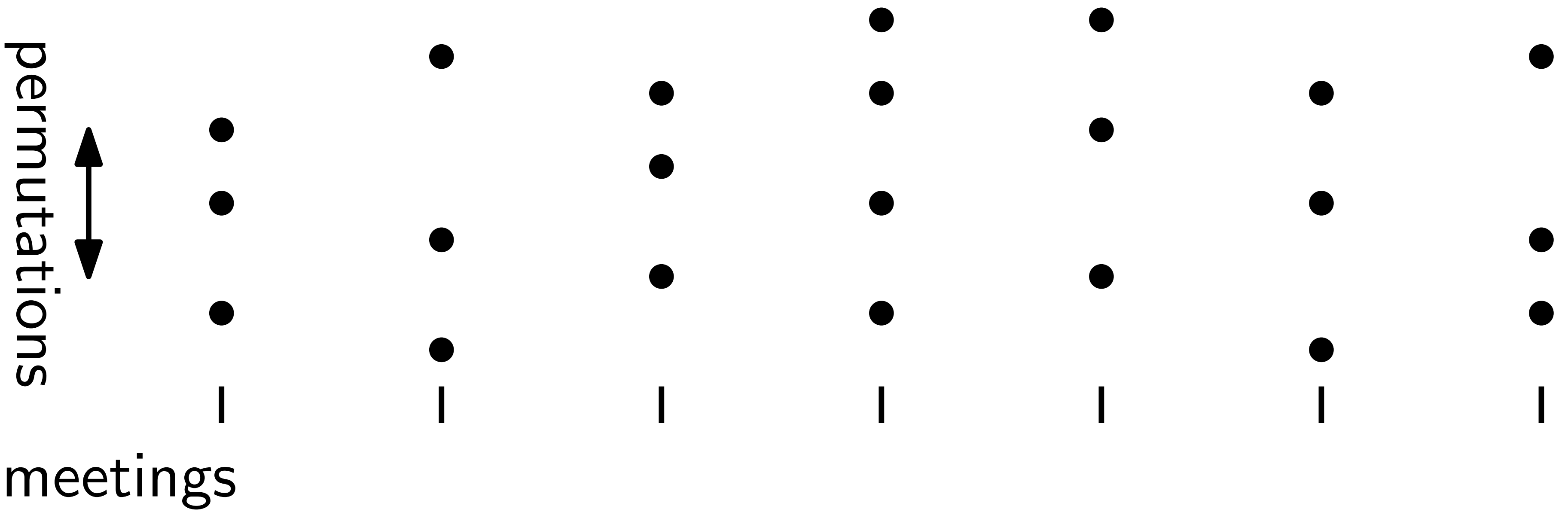
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path
- runtime: $O(k!^{2}n)$

Exact Algorithm

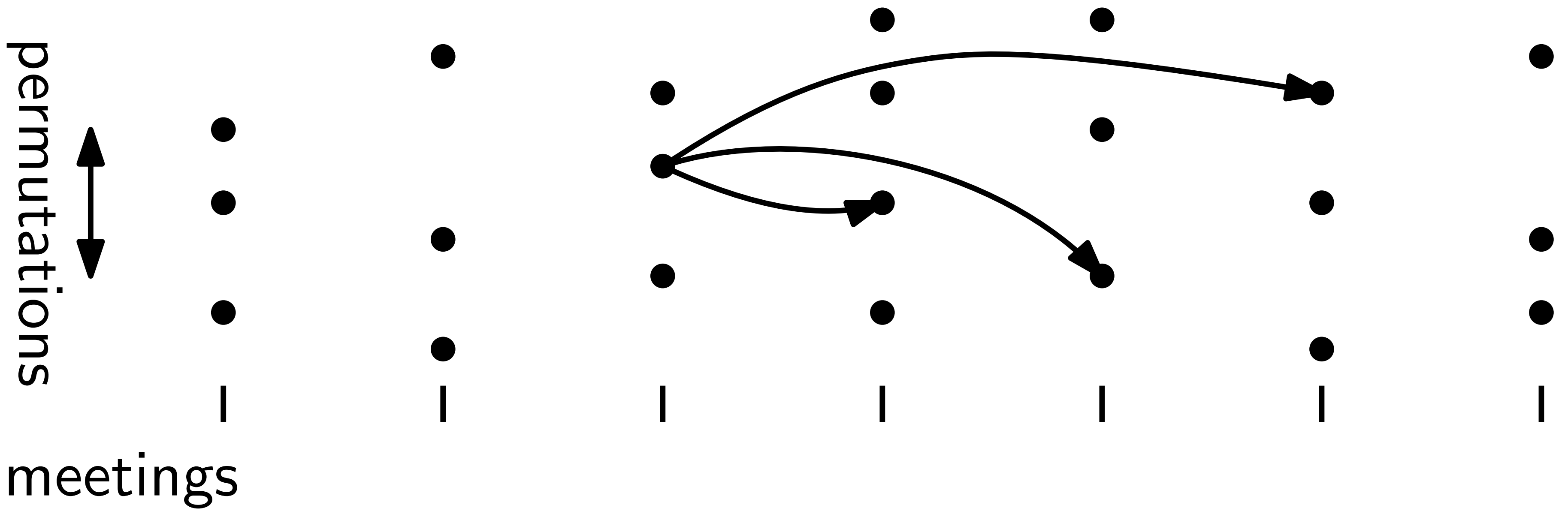
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path
- runtime: $O(k!^{\mathbf{2}} n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

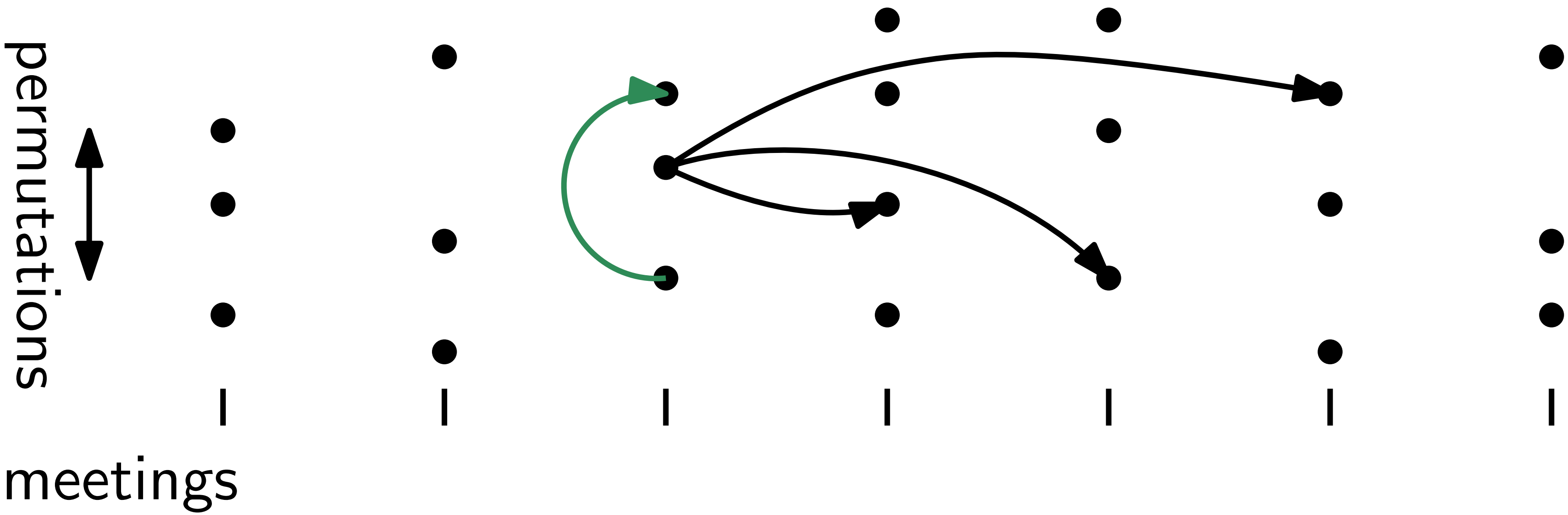
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path
- runtime: $O(k!^{\mathbf{2}} n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

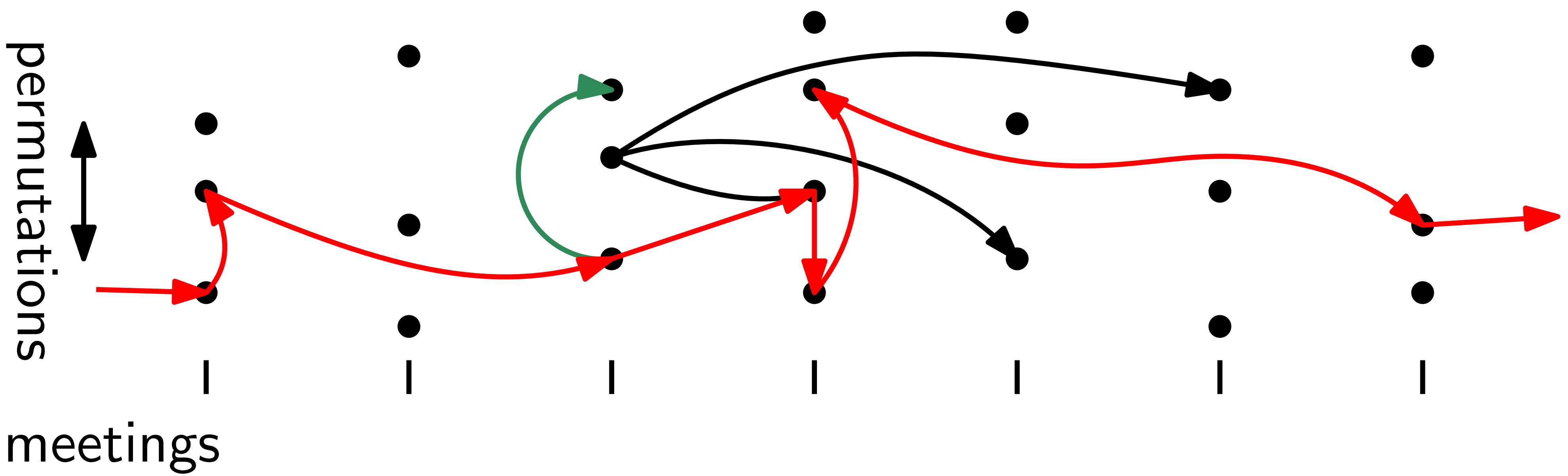
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path
- runtime: $O(k!^{\mathbf{2}} n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

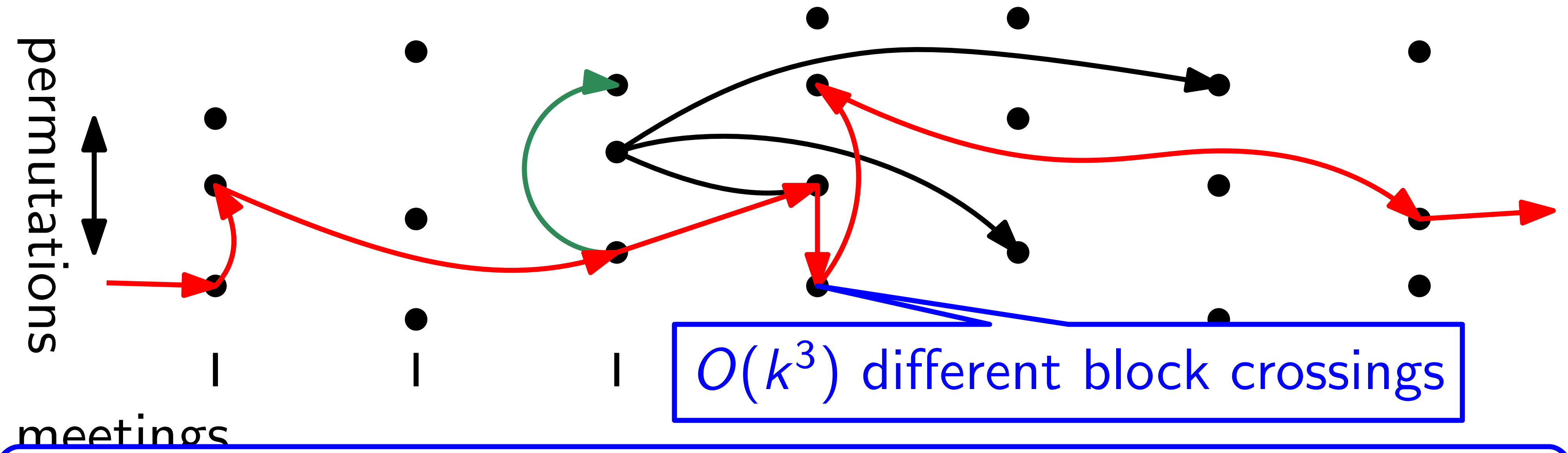
- first idea: modify FPT of Kostitsyna et al. for block crossings



- find minimum-cost path
- runtime: $O(k!^{\color{red}2} n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



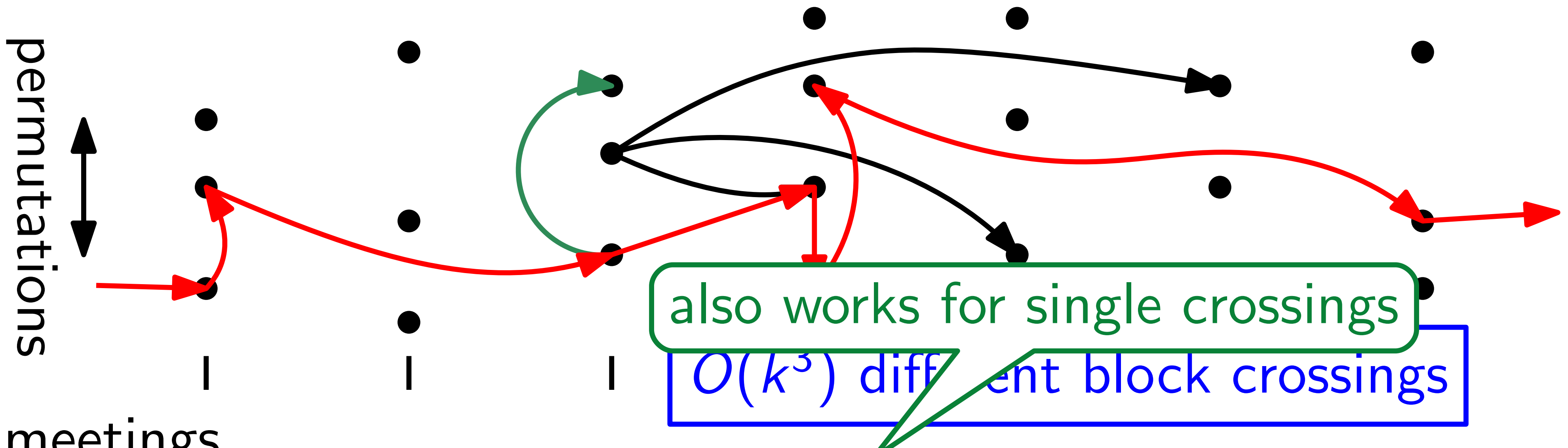
Theorem: We can minimize block crossings in $O(k!k^3n)$ time and $O(k!kn)$ space.

- runtime: $O(k!^{\mathbf{2}}n)$

- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block crossings



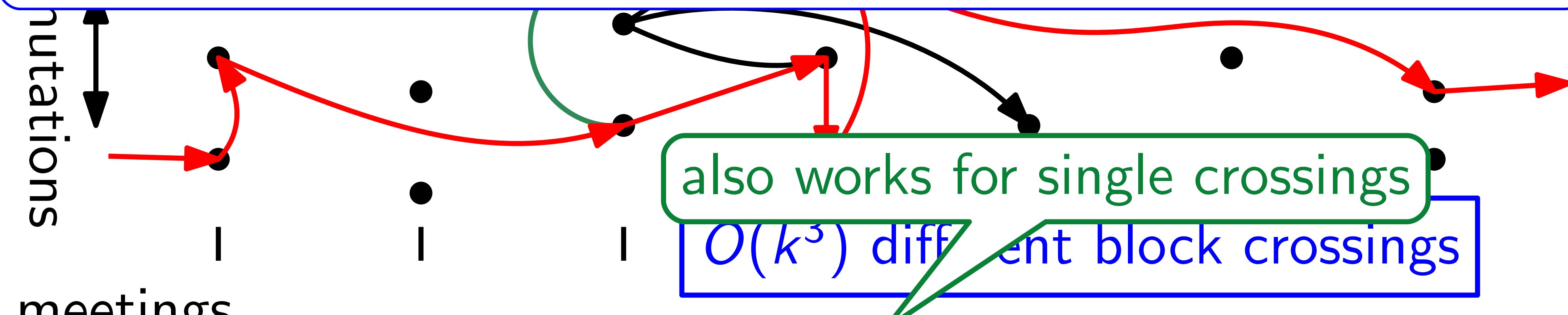
Theorem: We can minimize block crossings in $O(k!k^3n)$ time and $O(k!kn)$ space.

- runtime: $O(k!^{2}n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

Exact Algorithm

- first idea: modify FPT of Kostitsyna et al. for block

Alternative: Can minimize block crossings in $O(k!k^{\beta}(\beta + kn))$ time and $O(\beta k)$ space, where $\beta =$ opt. #block crossings



also works for single crossings

$O(k^3)$ diff ent block crossings

Theorem: We can minimize block crossings in $O(k!k^3 n)$ time and $O(k!kn)$ space.

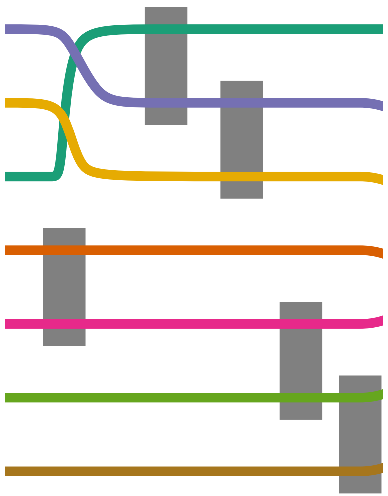
- runtime: $O(k!^{\mathbf{2}} n)$
- new idea: 1 edge $\leftrightarrow$ block crossing

2-Character Meetings – A Greedy Algorithm

- only pairwise meetings
- single block crossing suffices to bring pair together

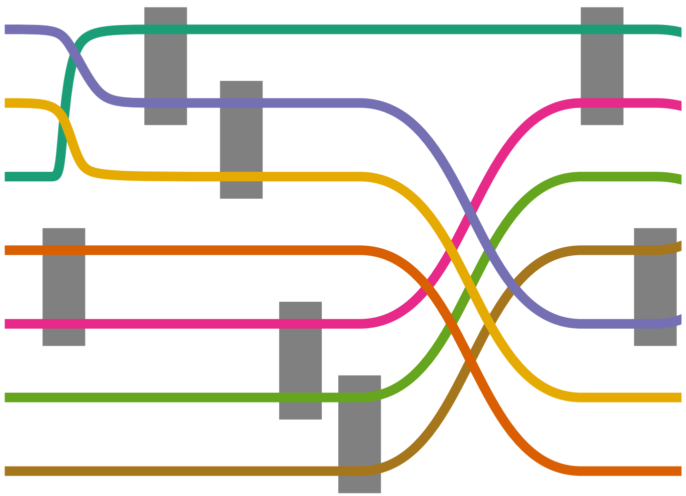
2-Character Meetings – A Greedy Algorithm

- only pairwise meetings
- single block crossing suffices to bring pair together



2-Character Meetings – A Greedy Algorithm

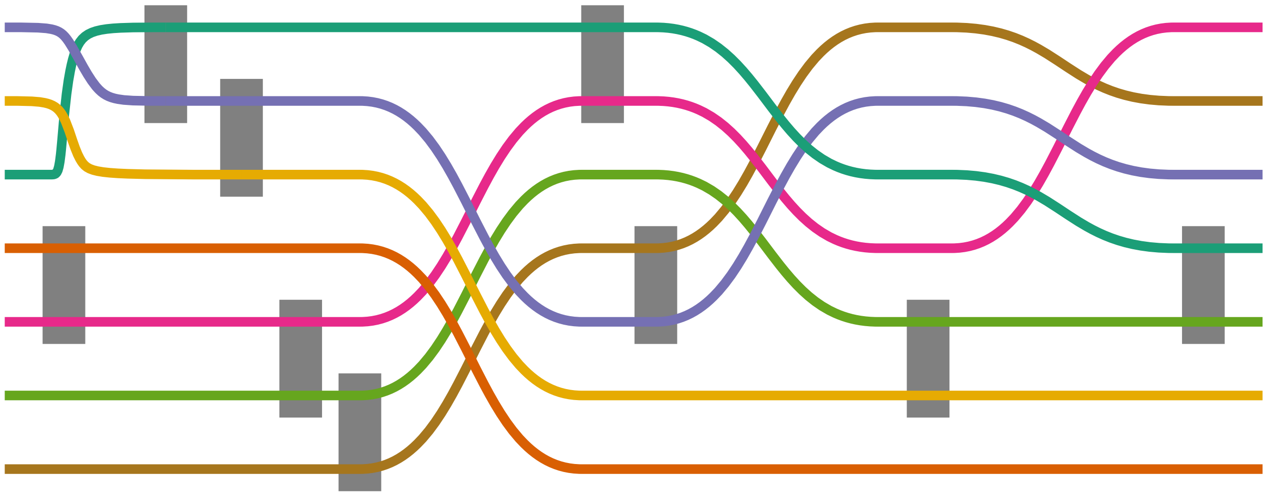
- only pairwise meetings
- single block crossing suffices to bring pair together



- single block crossing can support several new meetings

2-Character Meetings – A Greedy Algorithm

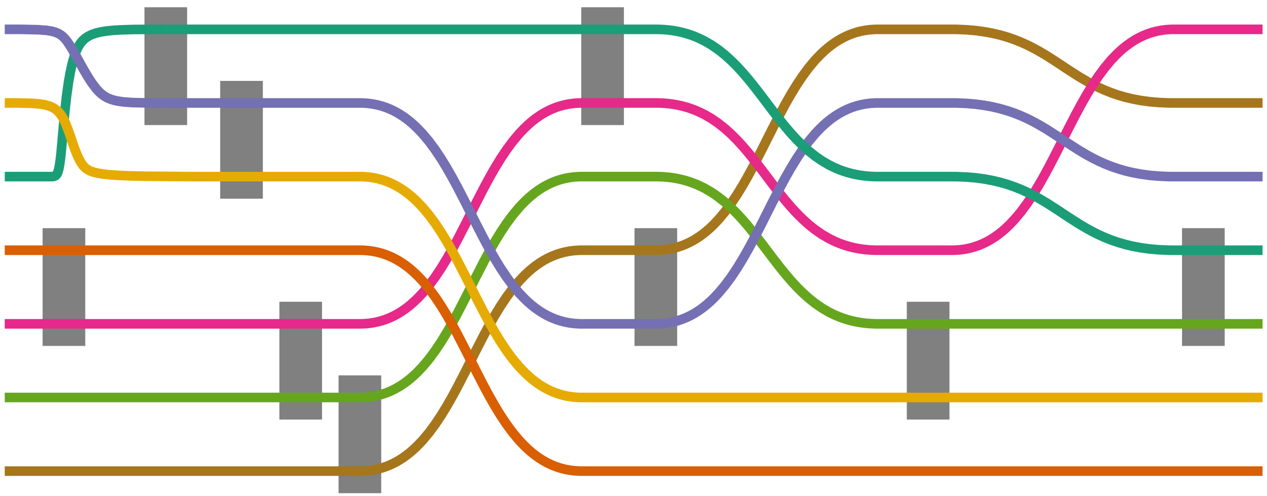
- only pairwise meetings
- single block crossing suffices to bring pair together



- single block crossing can support several new meetings
- greedily try to support largest prefix of future meetings with single block crossing

2-Character Meetings – A Greedy Algorithm

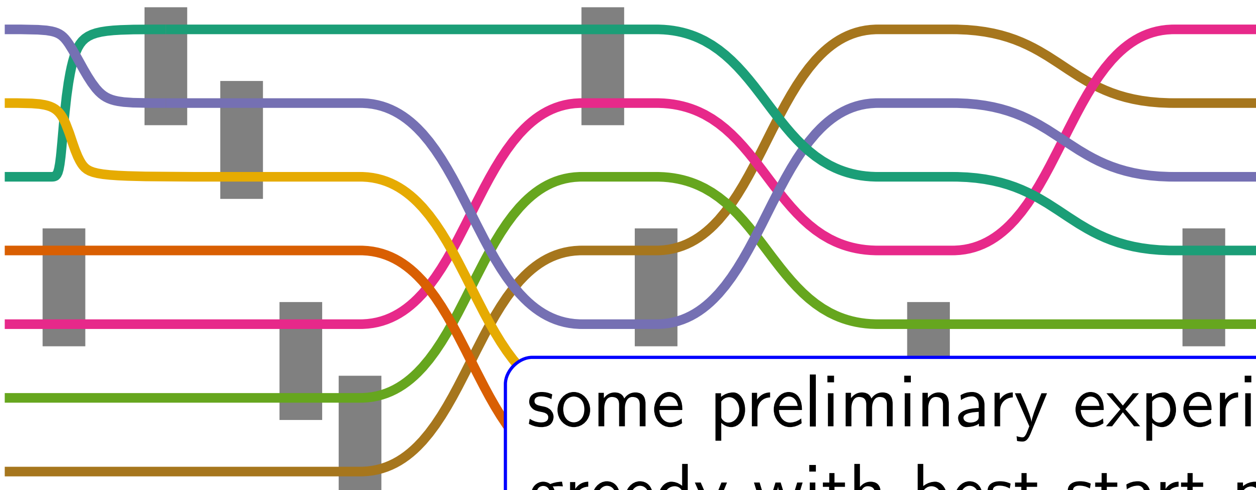
- only pairwise meetings
- single block crossing suffices to bring pair together



- single block crossing can support several new meetings
- greedily try to support largest prefix of future meetings with single block crossing
- $O(kn)$-time algorithm
- use random or best start permutation

2-Character Meetings – A Greedy Algorithm

- only pairwise meetings
- single block crossing suffices to bring pair together



- single block crossir
- greedily try to supp
  with single block c
- $O(kn)$-time algorithm
- use random or best start permutation

some preliminary experiments; e.g.:
greedy with best start permutation
for $k = 5, n = 12$:
56% opt., 38% + 1bc, 5% + 2bc,
1% + 3bc

Conclusion

- can identify crossing-free solution
- new exact algorithms
- minimizing block crossings is hard
- approximation algorithm
- greedy heuristic for pairwise meetings

Conclusion

- can identify crossing-free solution
- new exact algorithms
- minimizing block crossings is hard
- approximation algorithm
- greedy heuristic for pairwise meetings

Open questions:

- generalize approximation / approximation for simple crossings?
- can greedy algorithm be generalized?

Conclusion

- can identify crossing-free solution
- new exact algorithms
- minimizing block crossings is hard
- approximation algorithm
- greedy heuristic for pairwise meetings

Open questions:

- generalize approximation / approximation for simple crossings?
- can greedy algorithm be generalized?

Thank you!