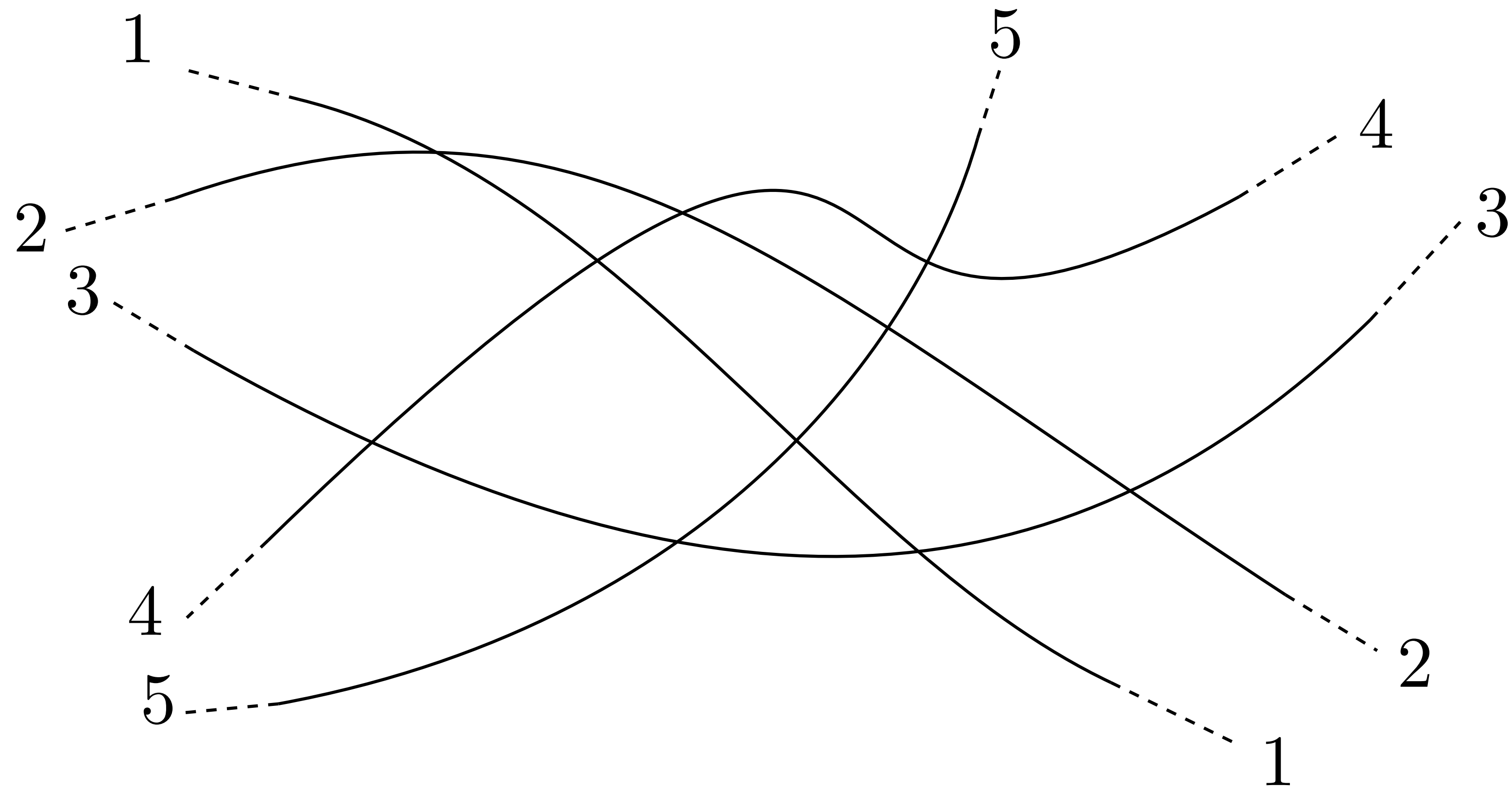
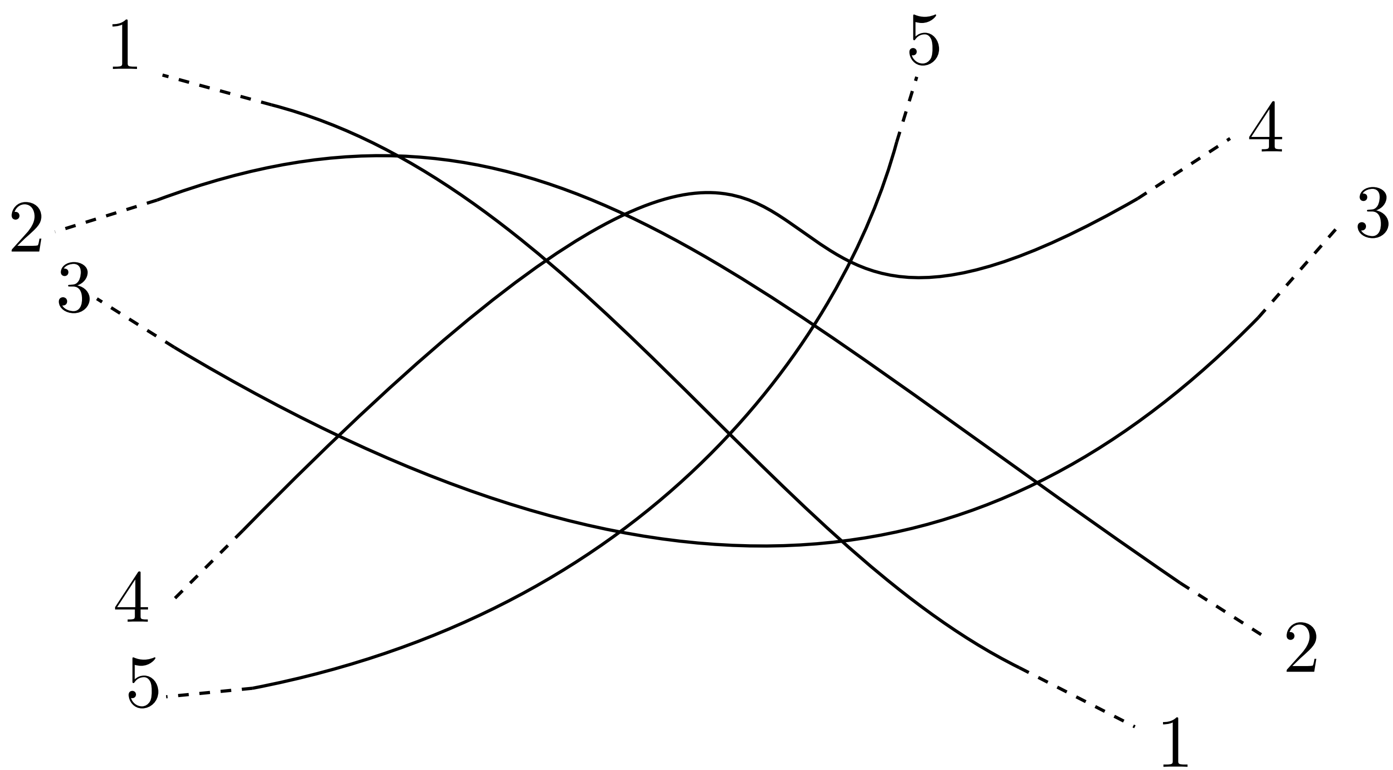
Enumeration and Counting of Pseudoline Arrangements

Günter Rote

Freie Universität Berlin



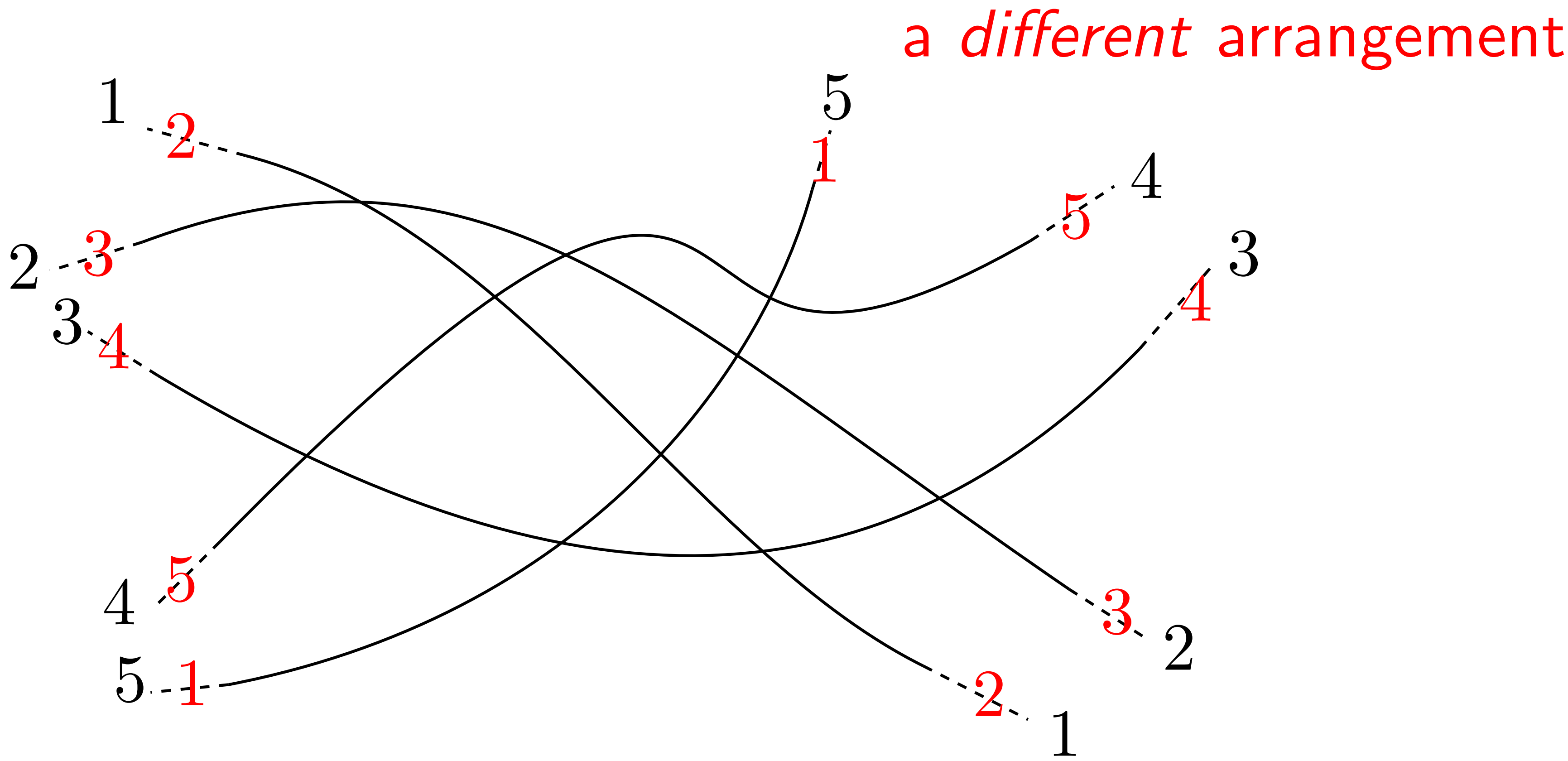
Pseudoline Arrangements



- n curves that go to infinity
- Two curves intersect exactly once, and they cross.

- *simple* pseudoline arrangements: no multiple crossings
- x-monotone curves

Pseudoline Arrangements



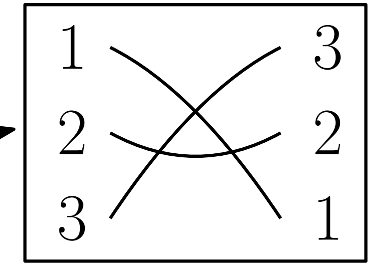
- n curves that go to infinity
- Two curves intersect exactly once, and they cross.

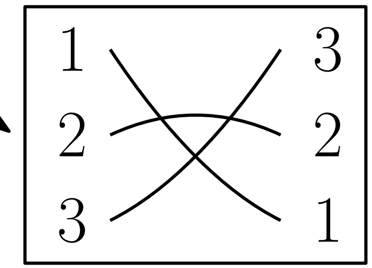
- *simple* pseudoline arrangements: no multiple crossings
- x-monotone curves

How many pseudoline arrangements?

Freie Universität Berlin

n	#PsA's with n pseudolines	
1	1	
2	1	
3	2	
4	8	
5	62	
6	908	
7	24698	
8	1232944	
9	112018190	
10	18410581880	
11	5449192389984	
12	2894710651370536	
13	2752596959306389652	
14	4675651520558571537540	[Yuma Tanaka, 2013]
15	14163808995580022218786390	
16	76413073725772593230461936736	[G. Rote, 2021]





OEIS A006245

How many pseudoline arrangements?

n	#PsA's with n pseudolines
1	1
2	1
3	2
4	8
5	62
6	908
7	24698
8	1232944
9	112018190
10	18410581880
11	5449192389984
12	2894710651370536
13	2752596959306389652
14	4675651520558571537540
15	14163808995580022218786390
16	76413073725772593230461936736

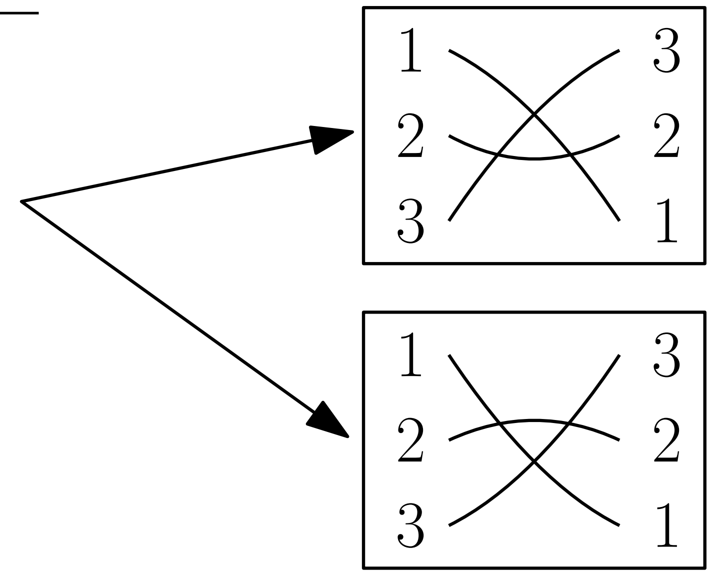
0.25 [Cortés Kühnast, Felsner, Scheucher 2023+]

$\geq 2^{\cancel{0.2083}n^2}$

[Dumitrescu, Mandal 2020]

$\leq 2^{0.6571n^2}$

[Felsner, Valtr 2012]



OEIS A006245

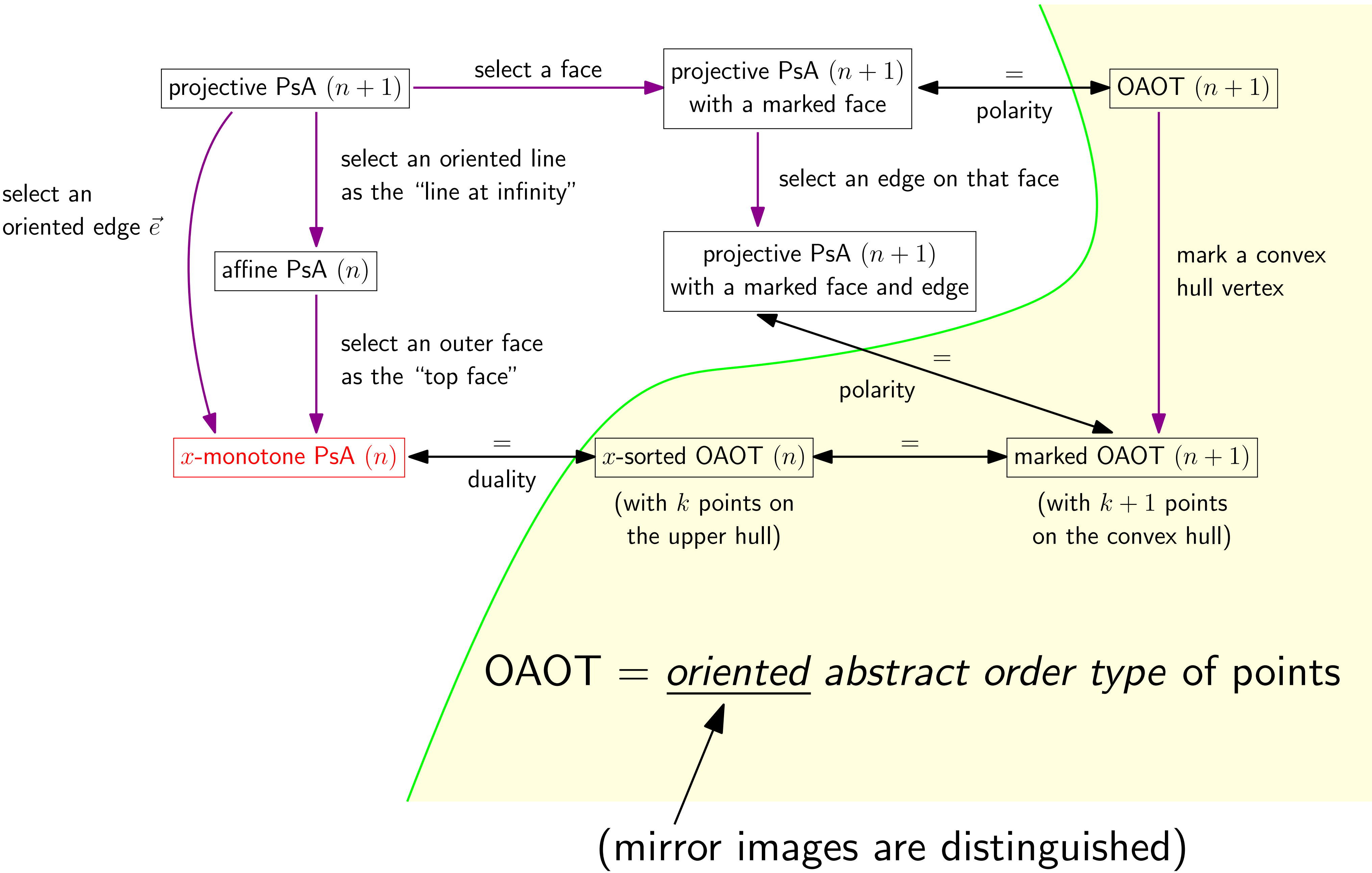
$n = 14, 15$: [Yuma Tanaka, 2013]

$n = 16$: [G. Rote, 2021]

Outline

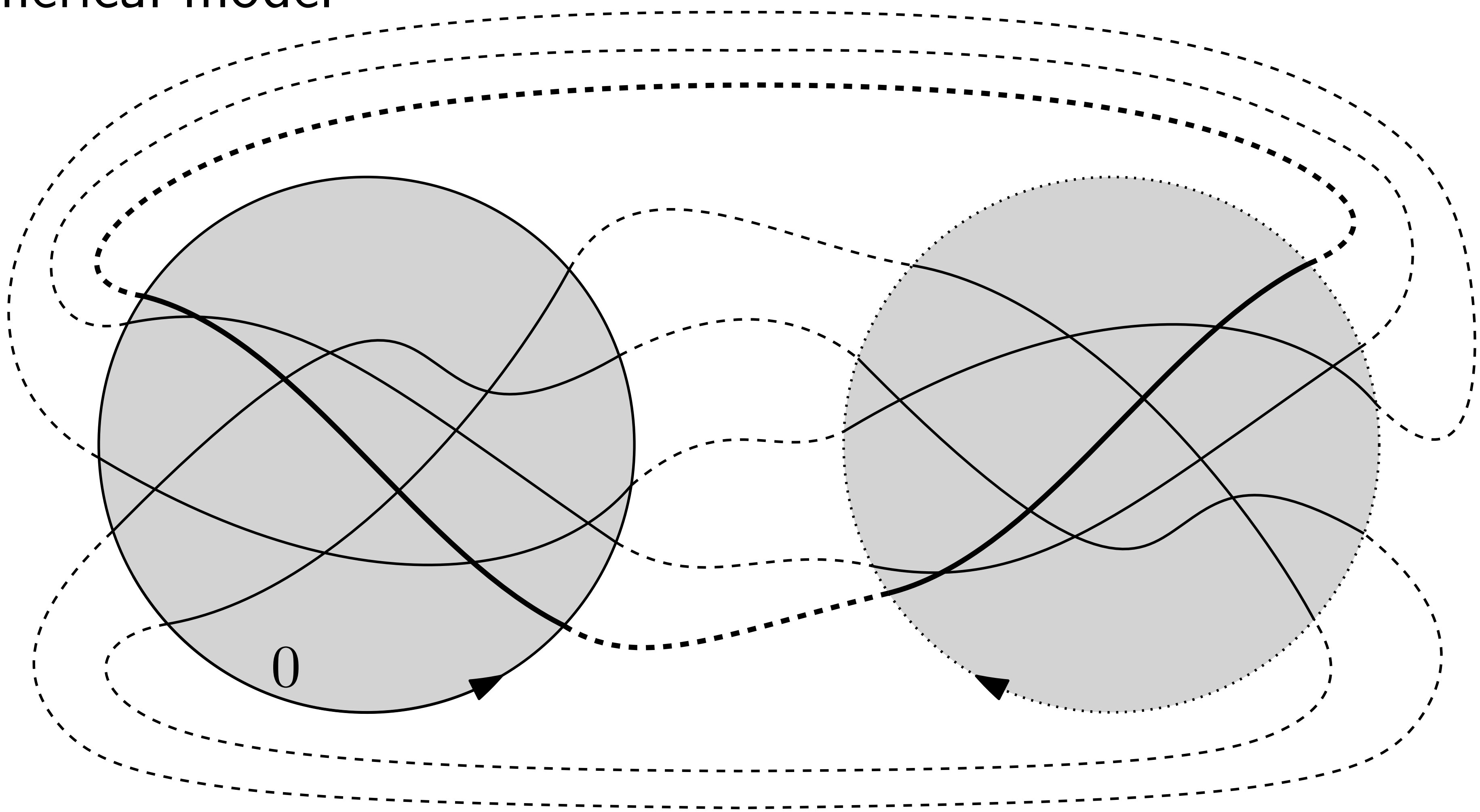
- Counting versus Enumeration
- What are we counting?
- 2-level-approach:
  Threading ℓ extra strands through
  a fixed arrangements of k pseudolines
- Partial pseudoline arrangements
- Sweep an arrangement (a bipolar orientation) with a rope

Related concepts



Projective pseudoline arrangements

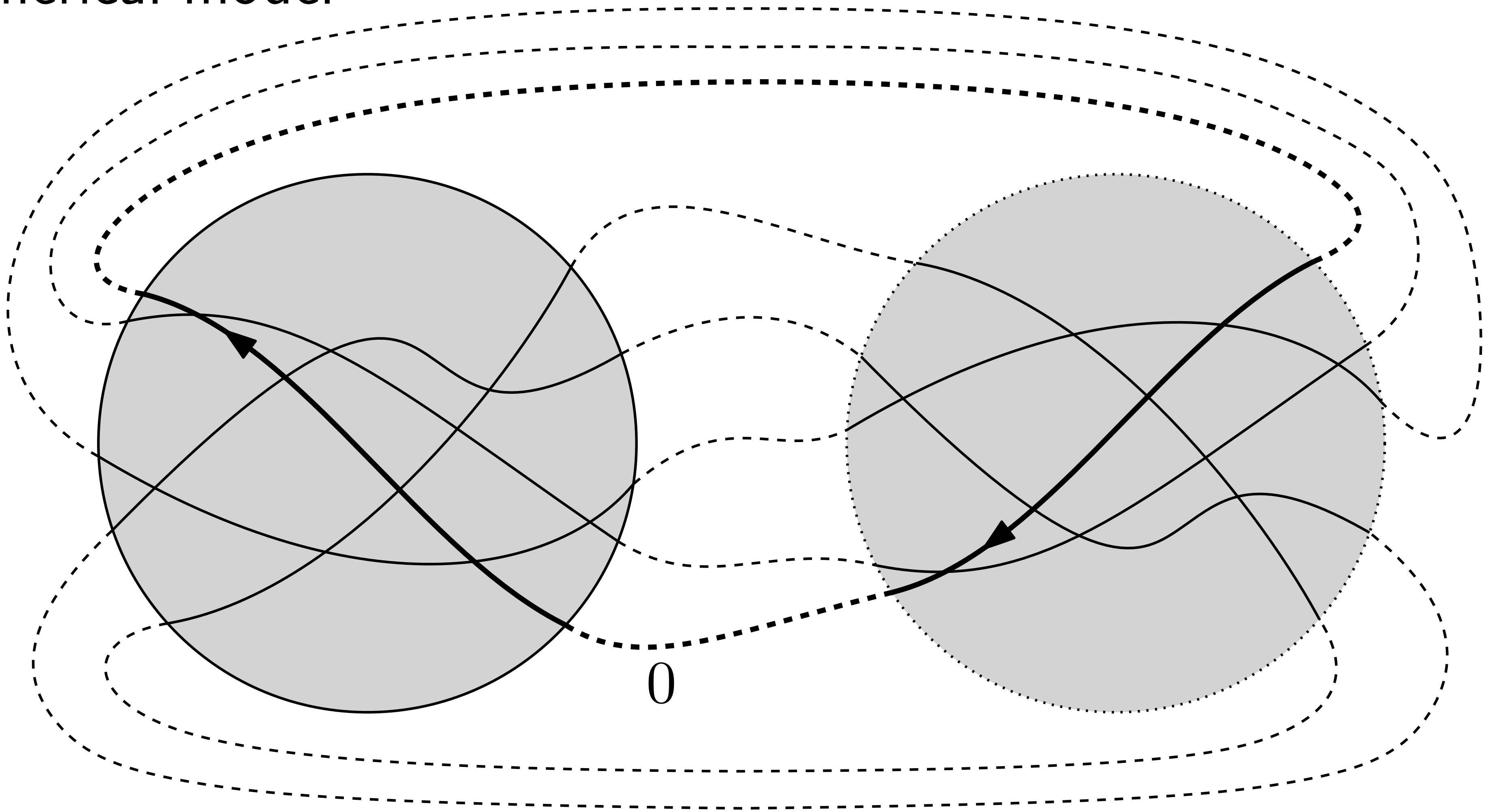
spherical model



Projective pseudoline arrangements

Freie Universität Berlin

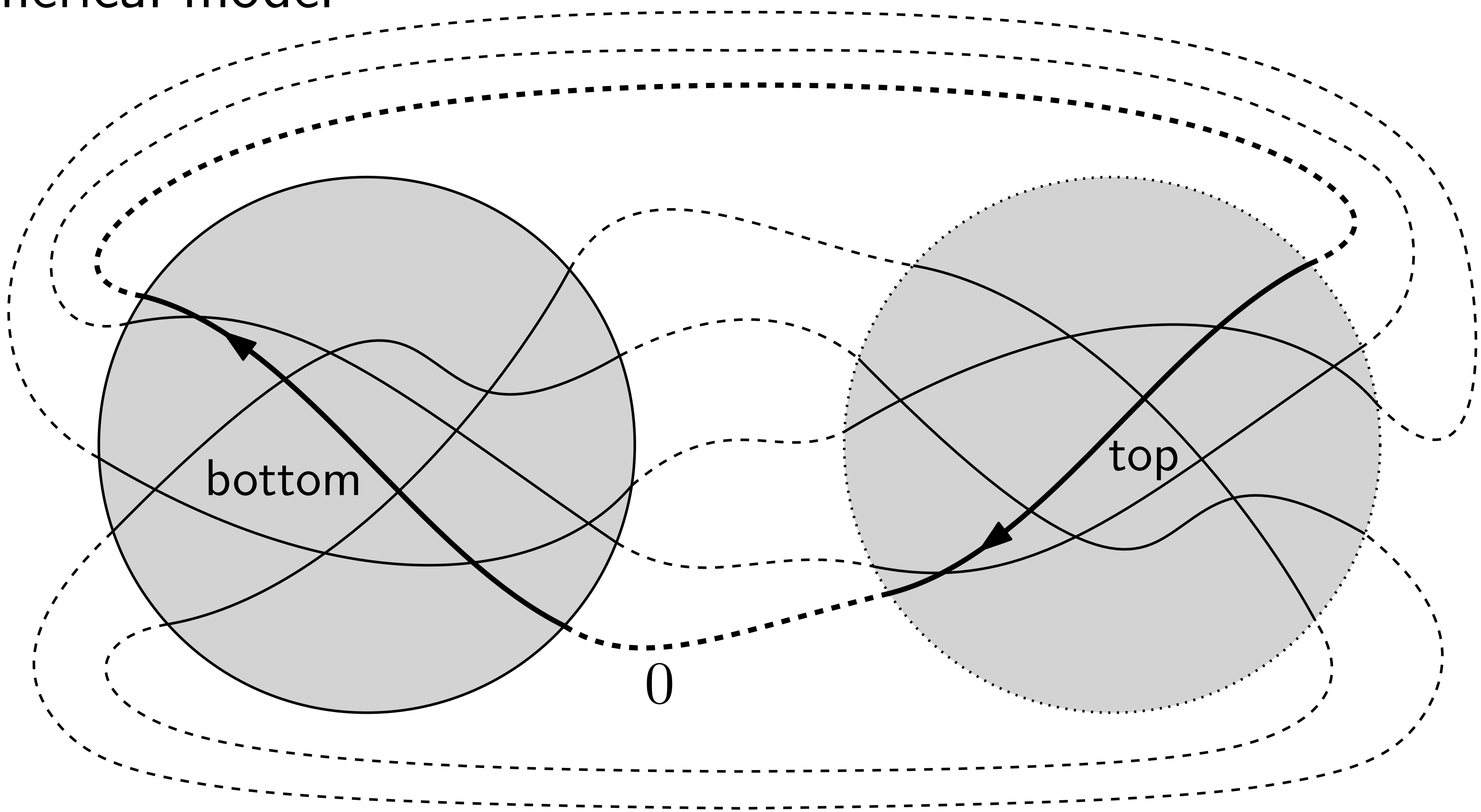
spherical model



- *affine* pseudoline arrangement

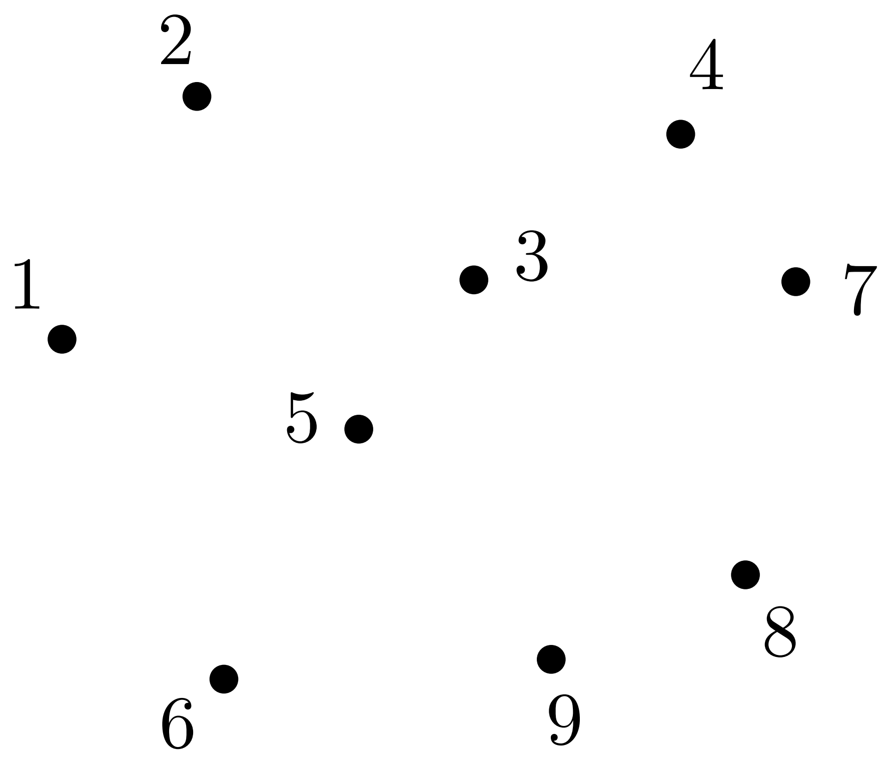
Projective pseudoline arrangements

spherical model

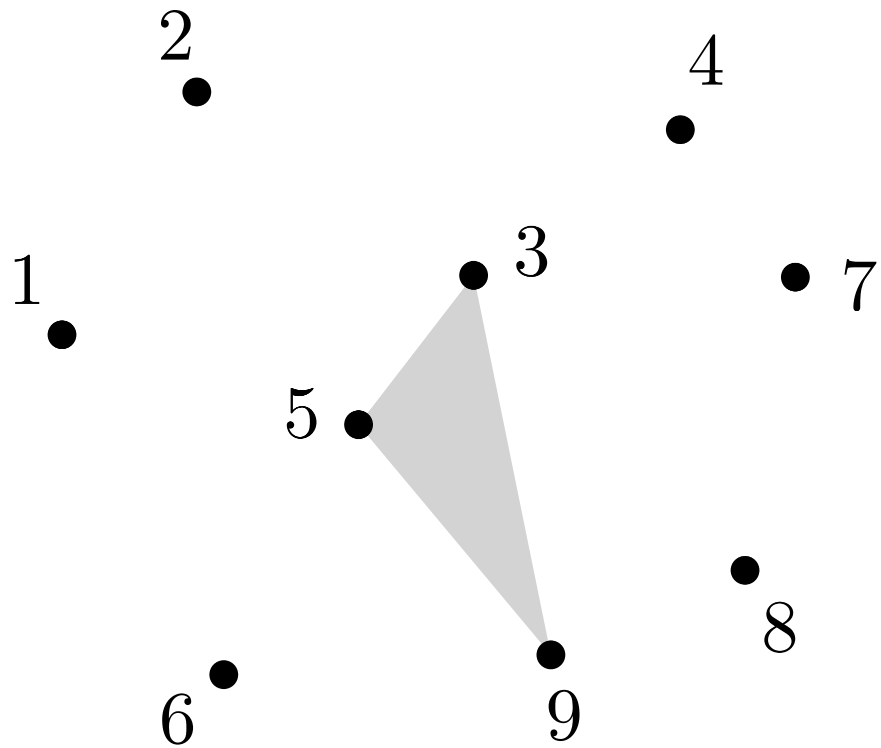


- *affine* pseudoline arrangement
- x-*monotone* pseudoline arrangement

(Abstract) order types of points



(Abstract) order types of points



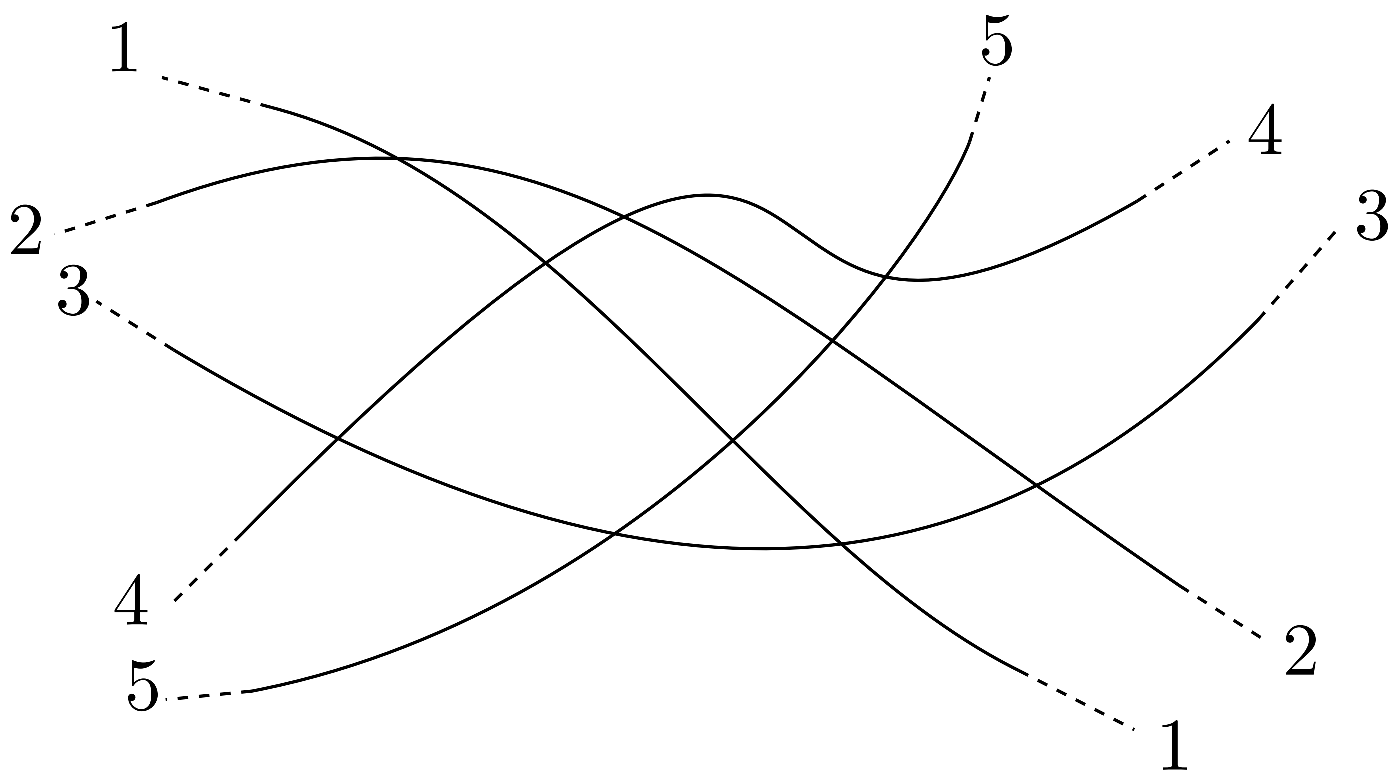
orientation

$593 = +$

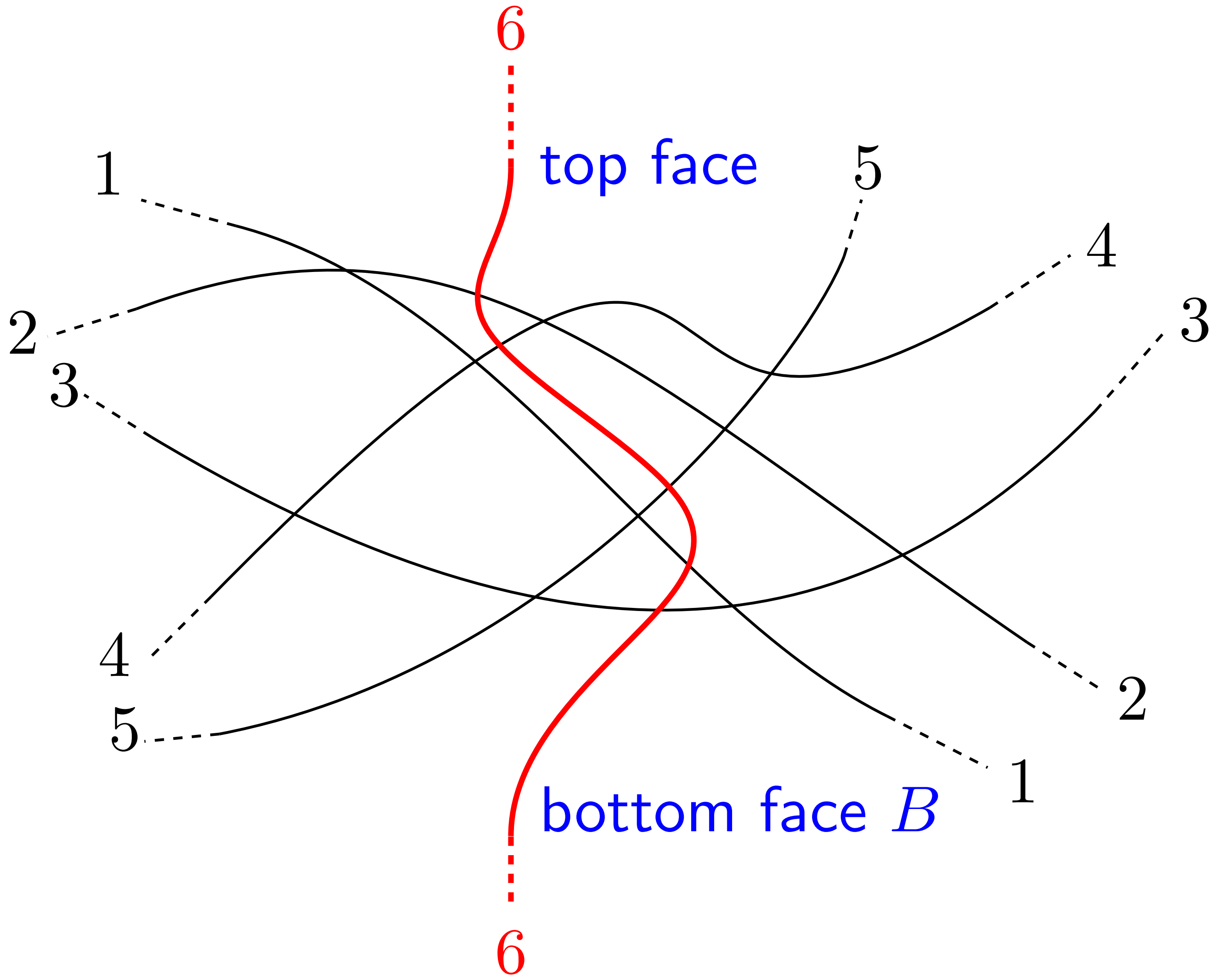
$359 = +$

$539 = -$

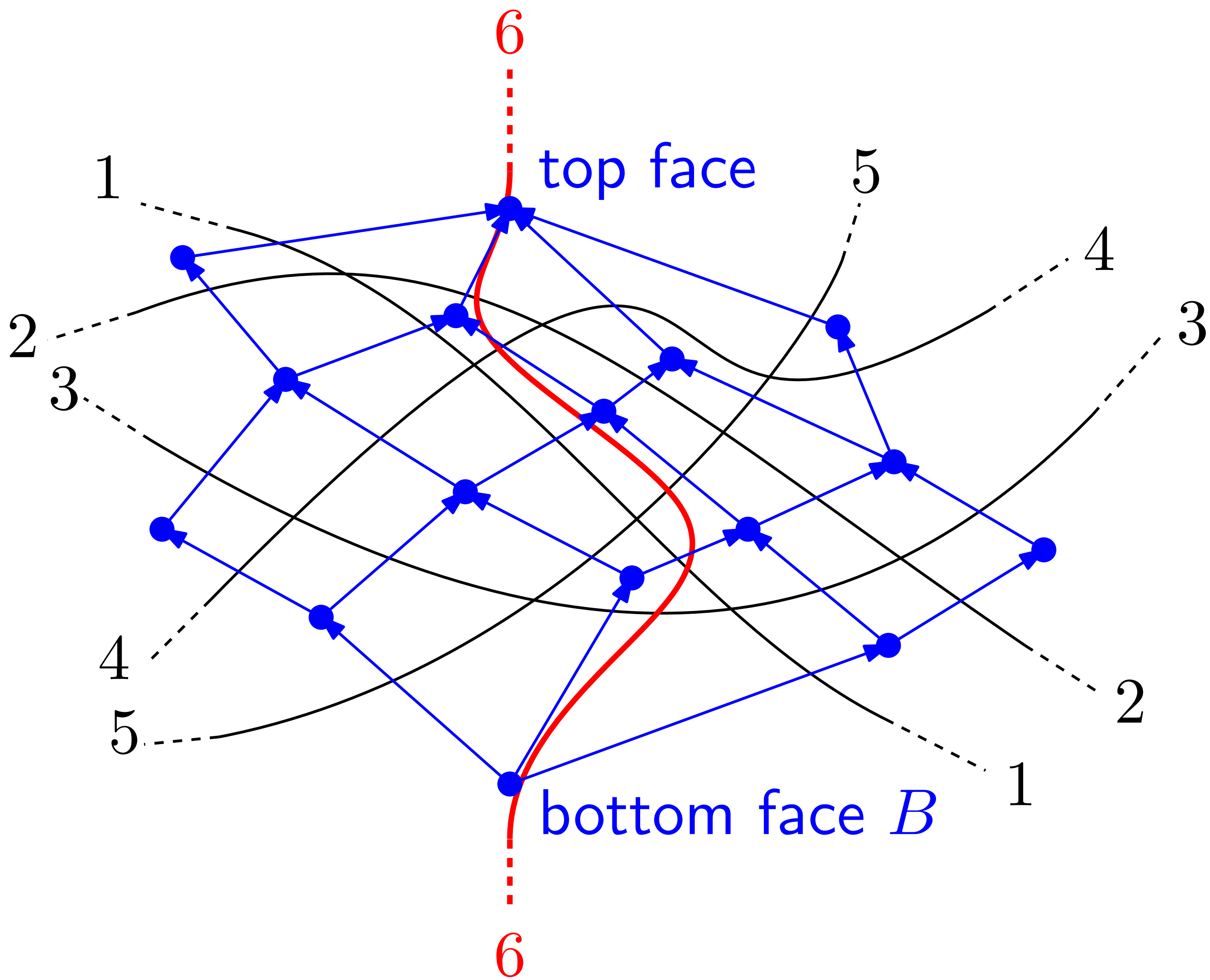
Inductive Enumeration of PsA's



Inductive Enumeration of PsA's

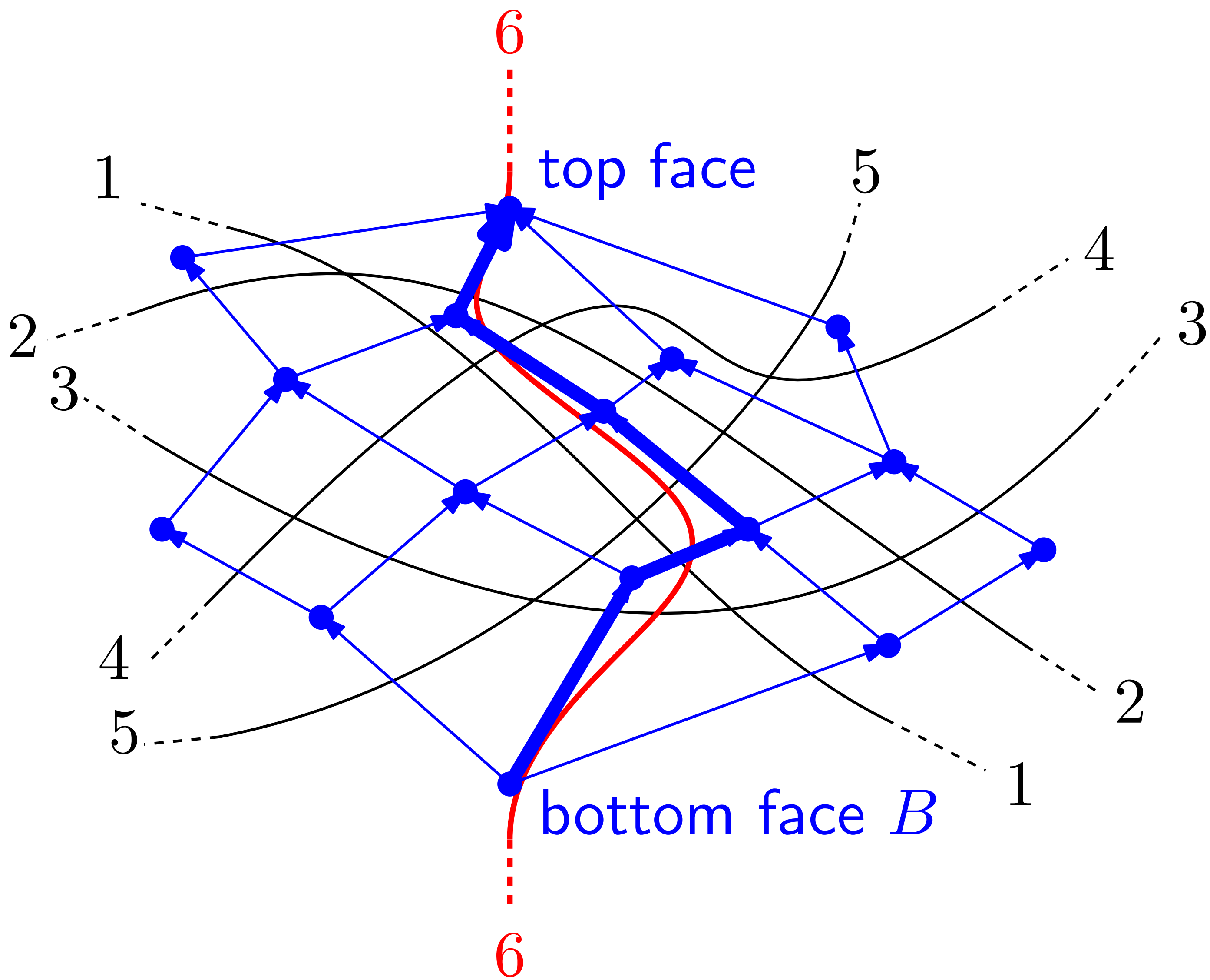


Inductive Enumeration of PsA's



pseudoline $n+1$ = path in the dual DAG

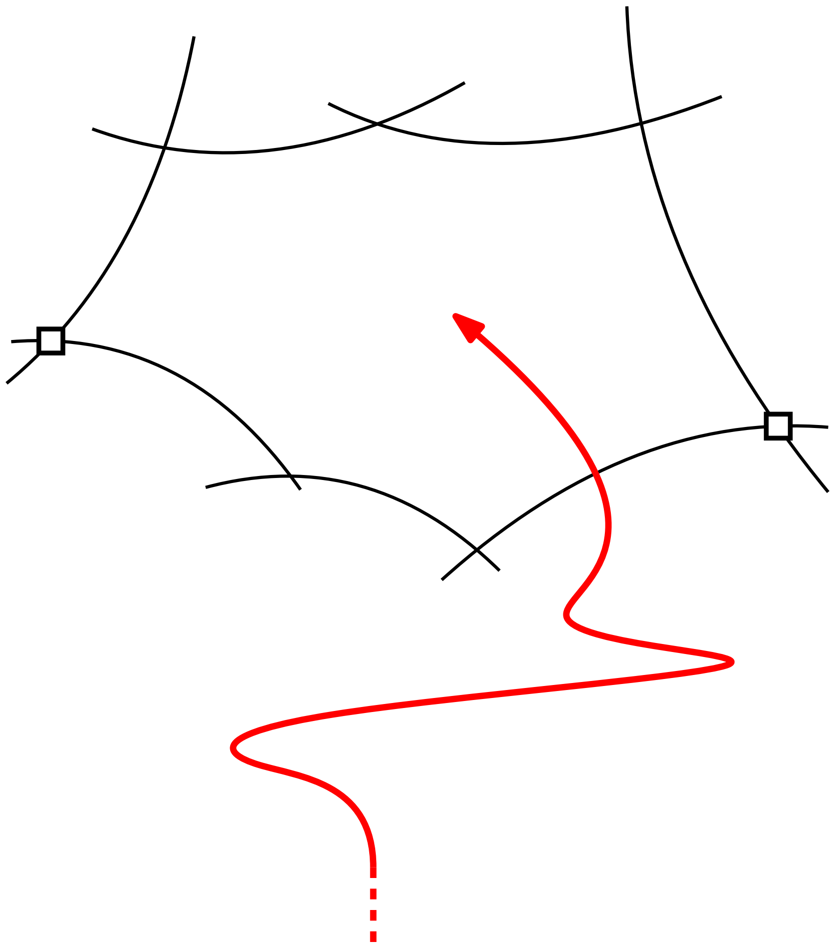
Inductive Enumeration of PsA's



pseudoline $n+1$ = path in the dual DAG

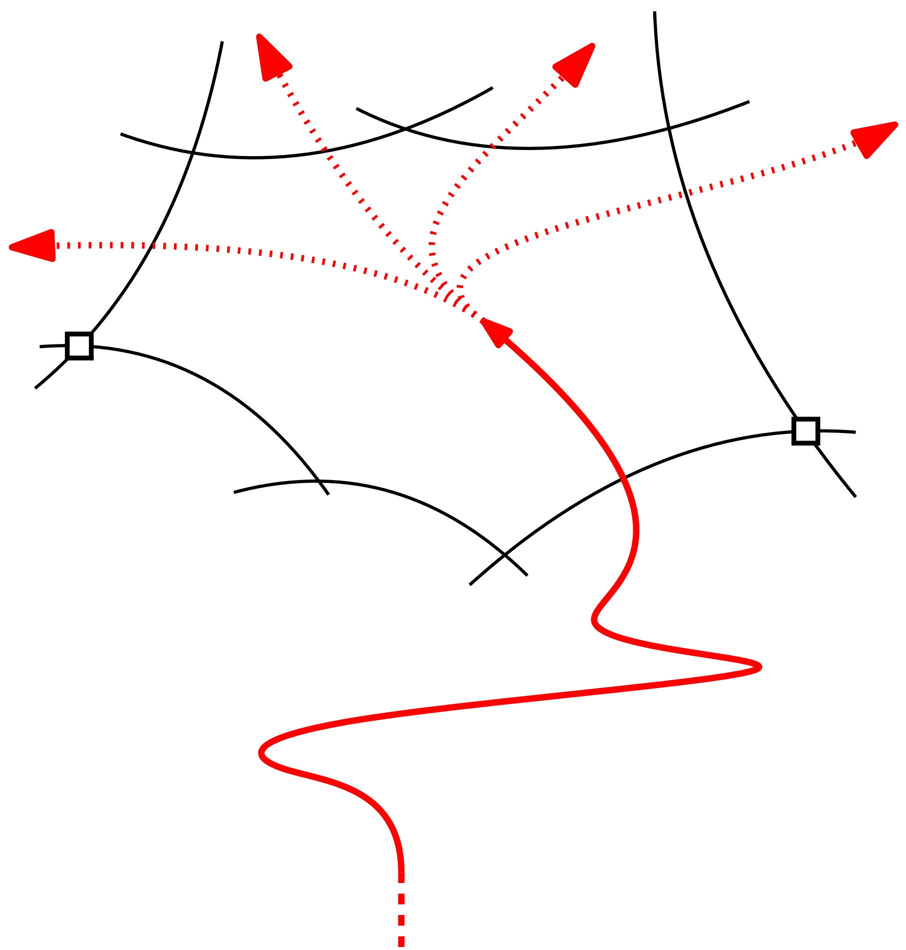
Inductive Enumeration of PsA's

Generation (enumeration) is straightforward. (No dead ends!)



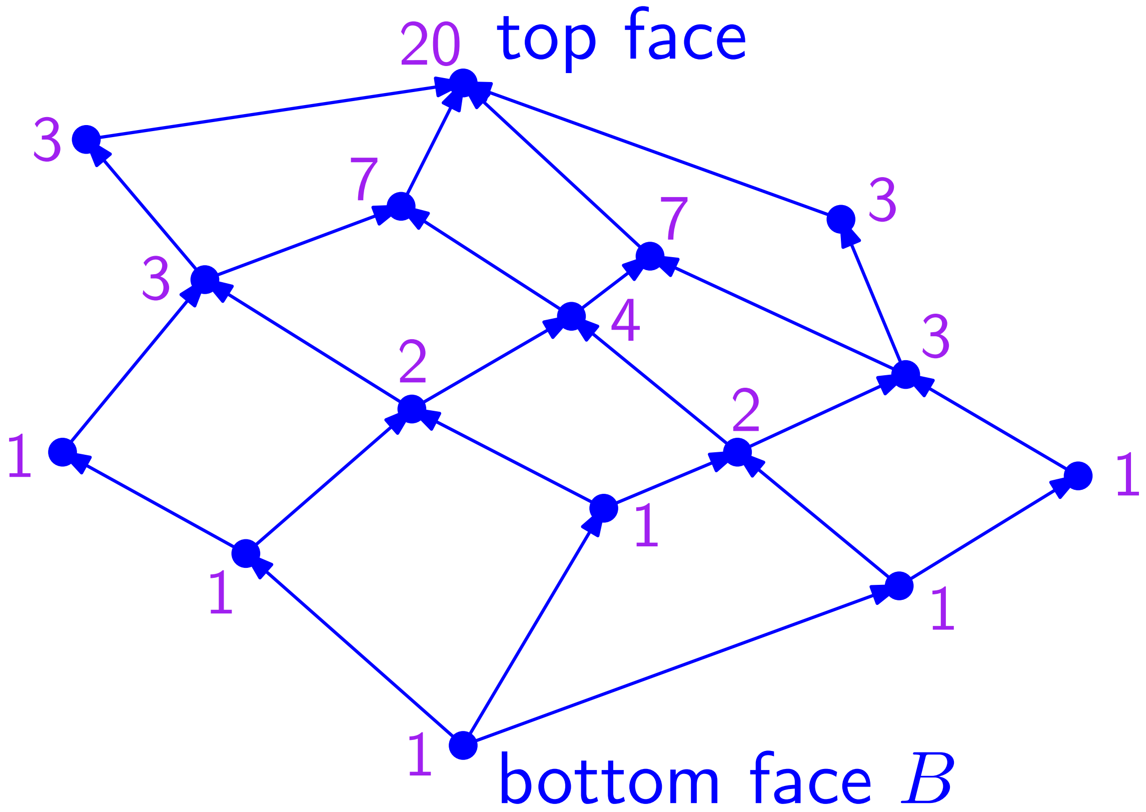
Inductive Enumeration of PsA's

Generation (enumeration) is straightforward. (No dead ends!)



Inductive Enumeration of PsA's

Counting is straightforward. (#paths from B in a DAG)



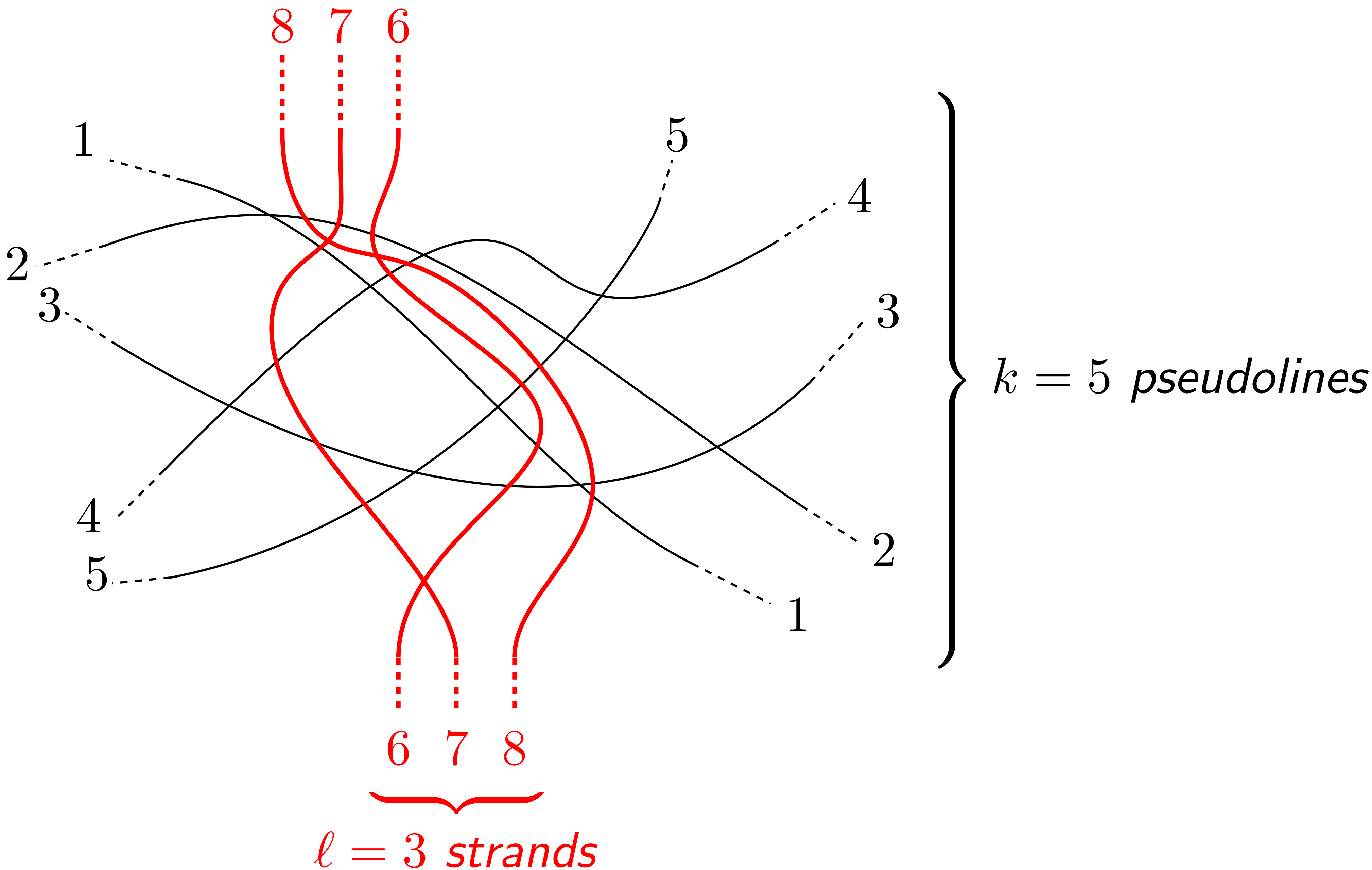
#paths $\leq 2.49^n$
[Felsner, Valtr 2012]

#paths can be as large as 2.076^n.
[O. Bílka 2010]

pseudoline $n+1$ = path in the dual DAG

Threading several pseudolines at once

Freie Universität Berlin



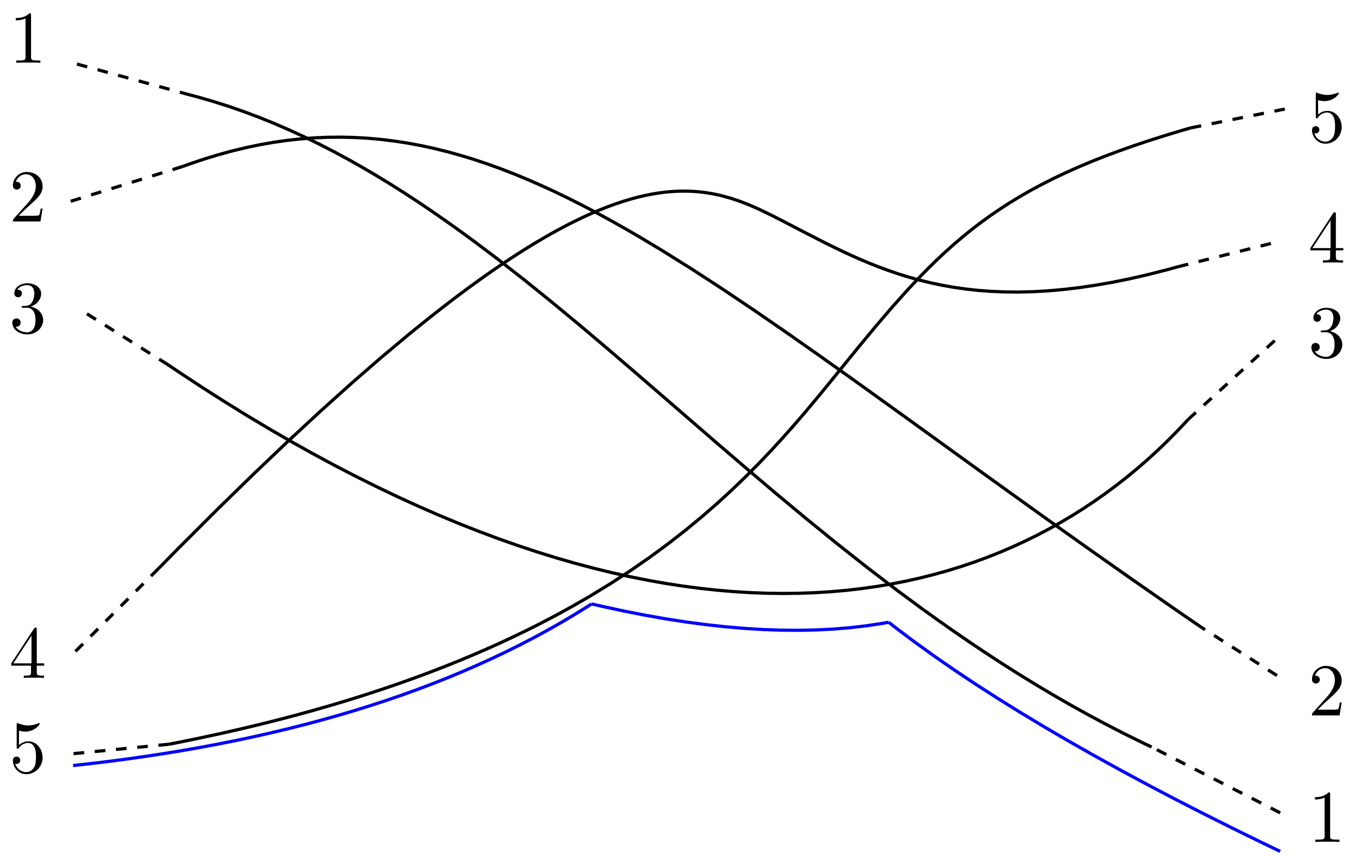
2-Level approach

Freie Universität Berlin

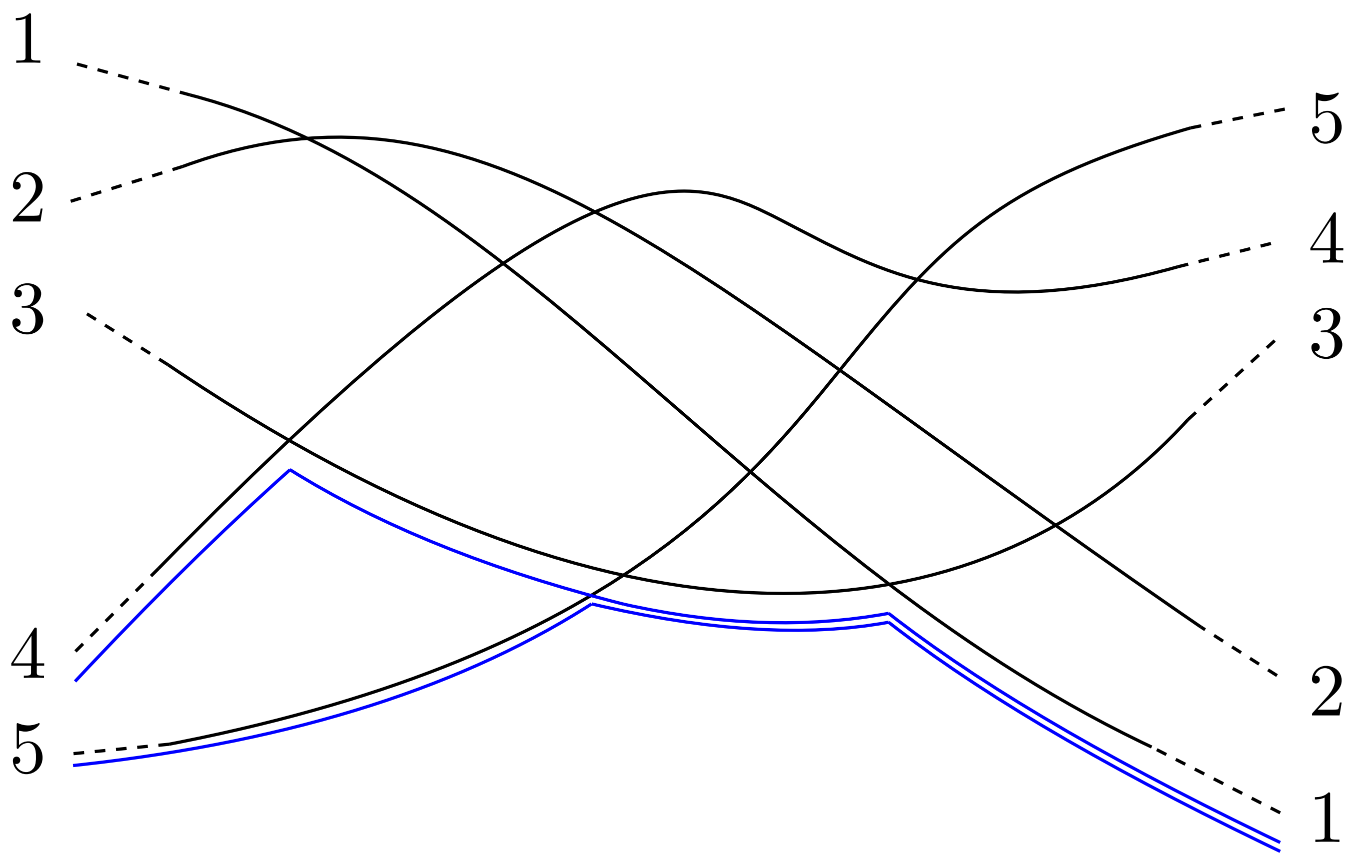
- *Enumerate* all arrangements of k pseudolines
- For each arrangement of k pseudolines:
  - *Count* the possibilites to thread ℓ extra strands

Preprocessing:
to deal with (partial) arrangements with ℓ strands fast

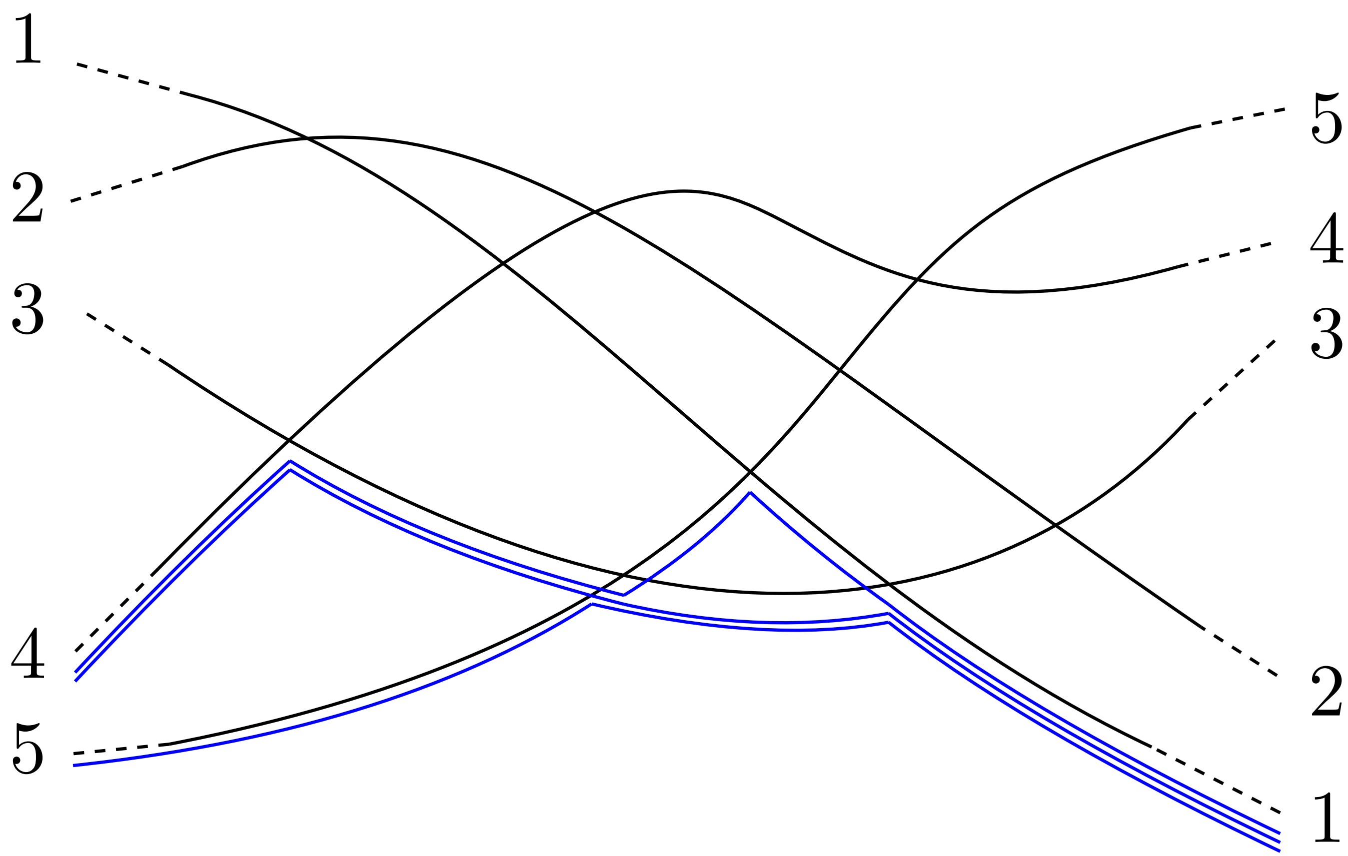
A sequence of ropes



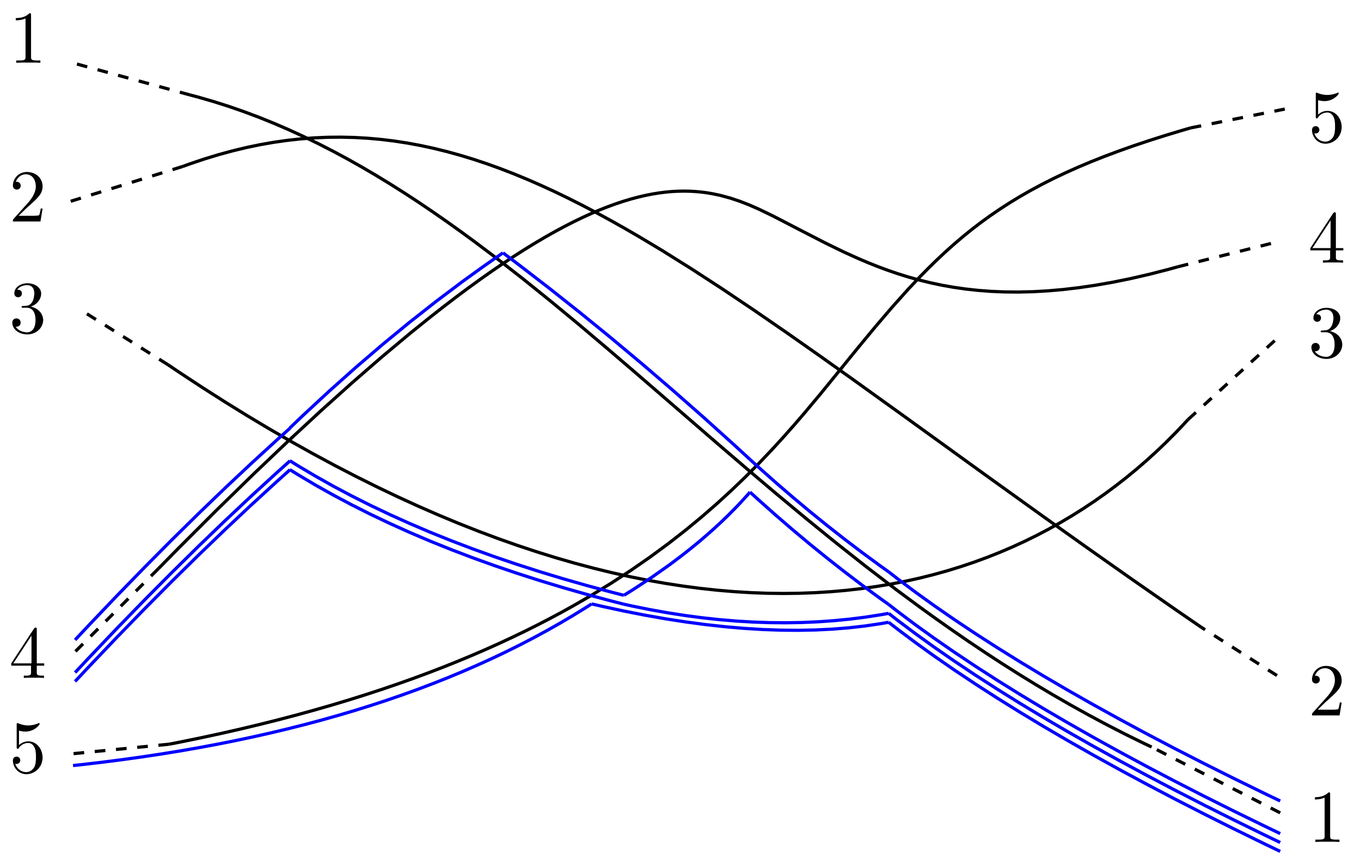
A sequence of ropes



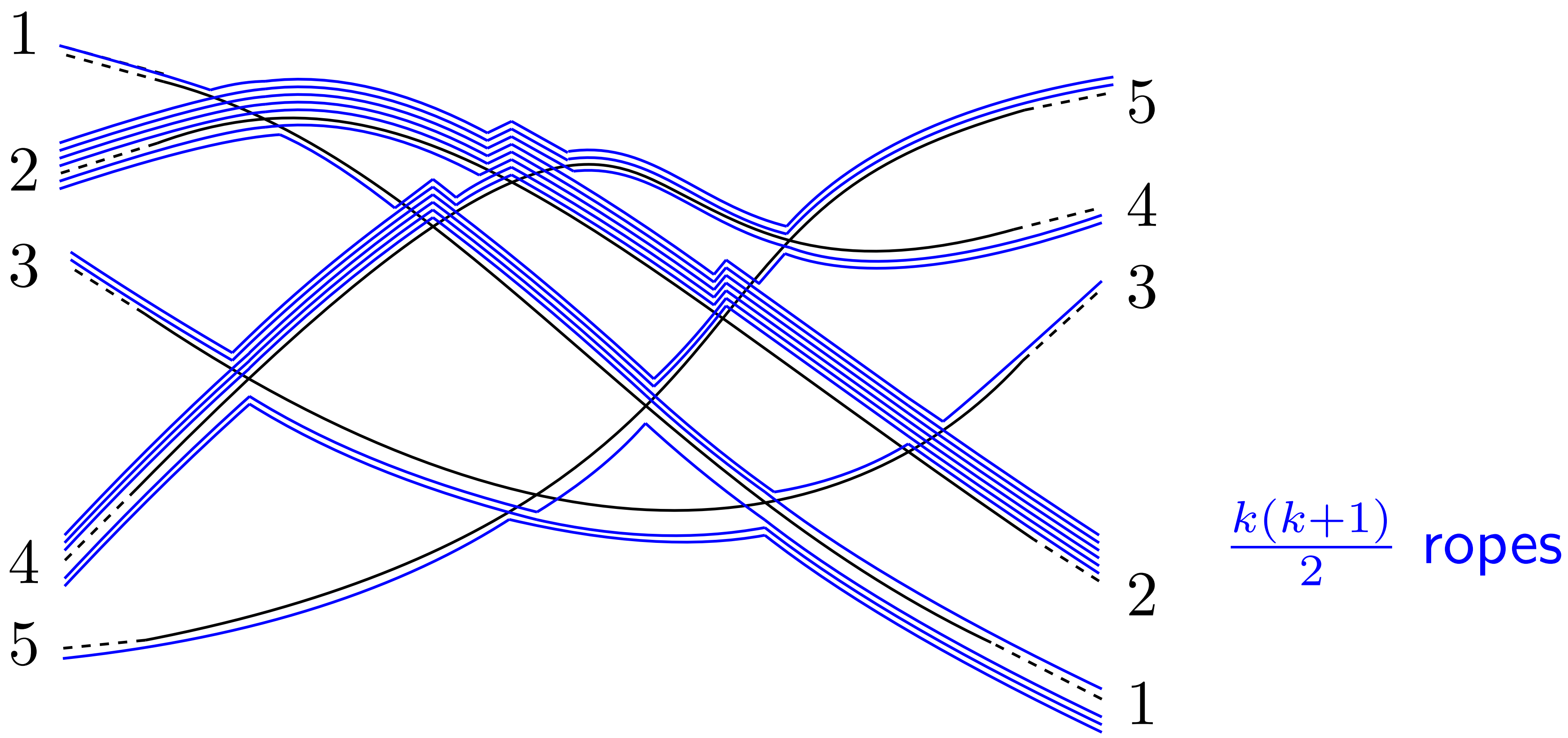
A sequence of ropes



A sequence of ropes



A sequence of ropes

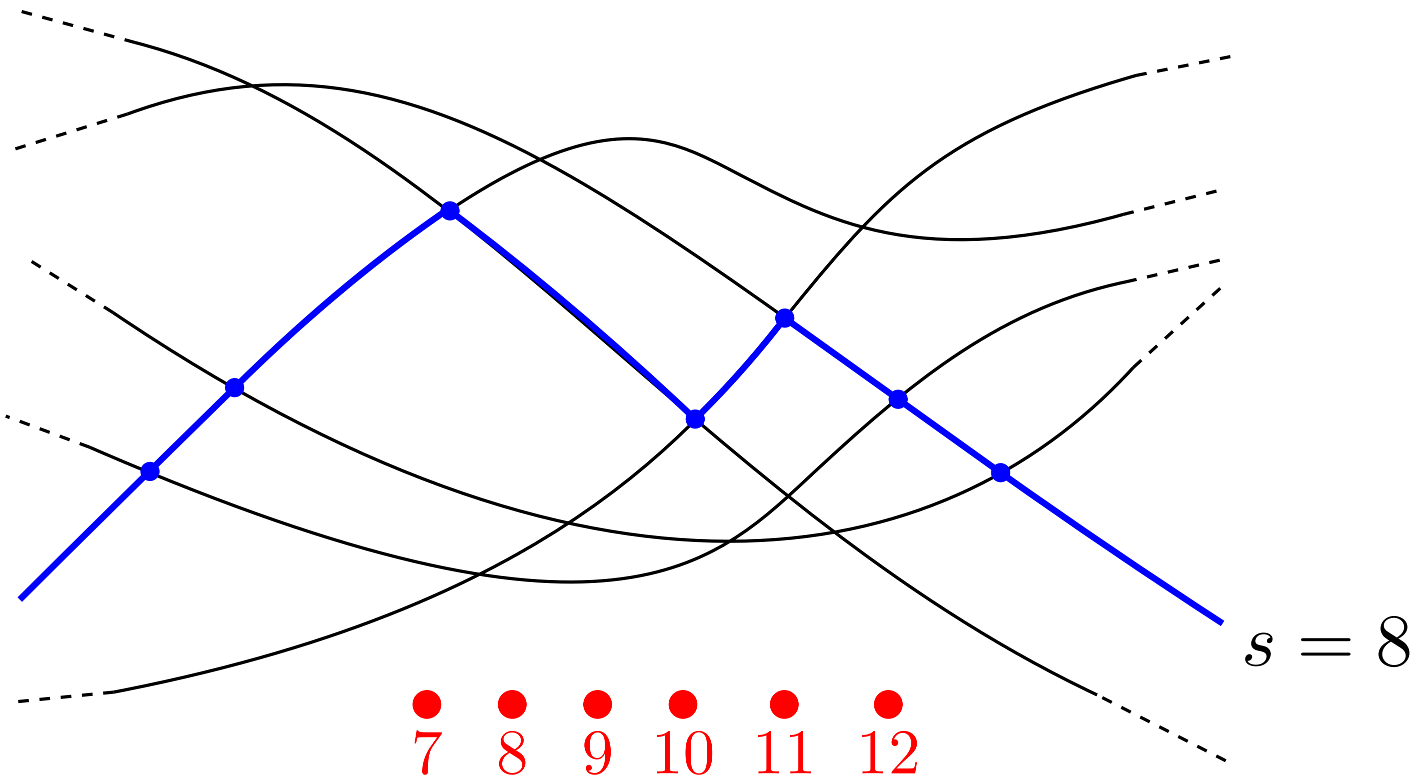


Take a fixed sweep by a sequence of ropes.

Dynamic programming

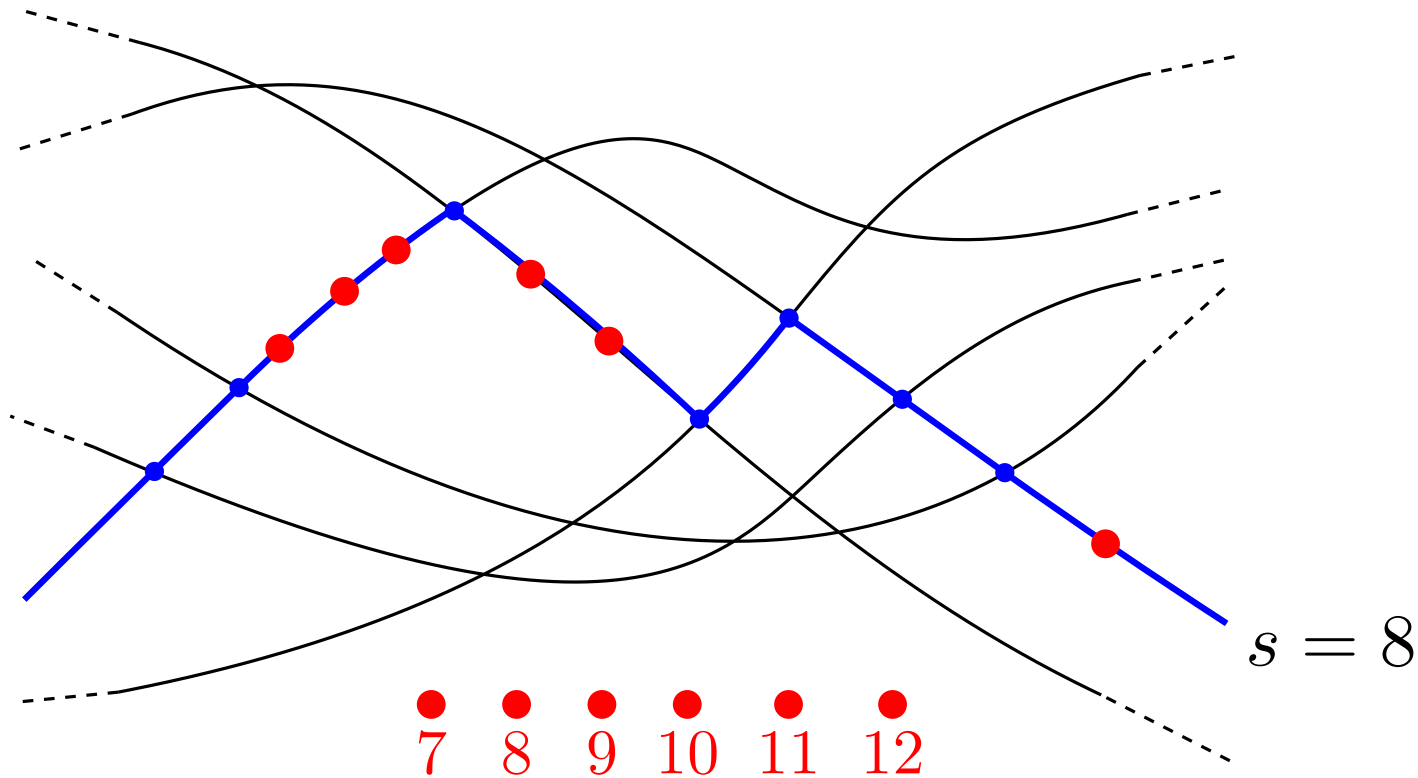
For each rope:

(s pieces)



Dynamic programming

For each rope:
(s pieces)



- For every distribution of the ℓ strands to the s pieces
- and for every permutation of the ℓ strands:

Store the number of possibilities to thread the ℓ strands from the bottom face to the rope.

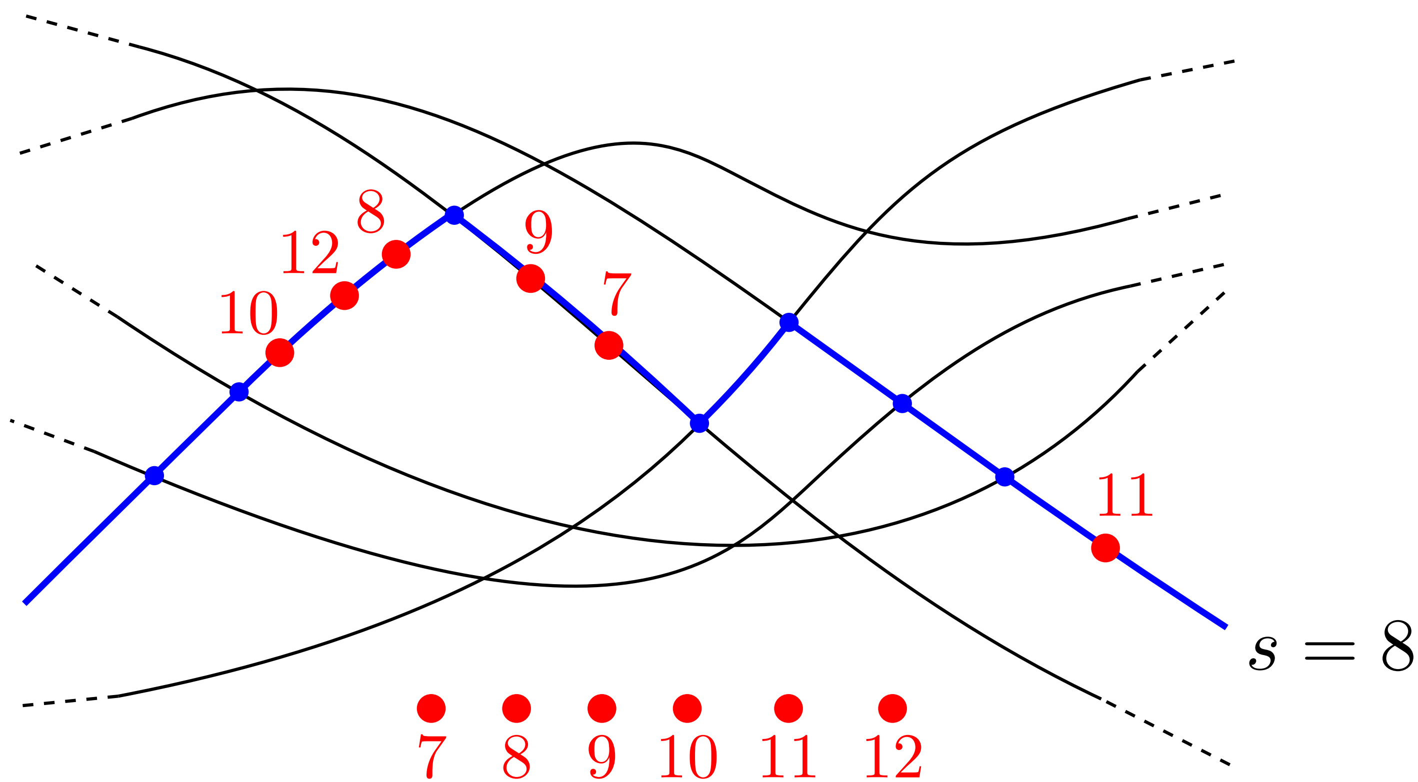
$\rightarrow s(s+1)(s+2)\ldots(s+\ell-1)$ entries

Dynamic programming

Freie Universität Berlin

For each rope:

(s pieces)



- For every distribution of the ℓ strands to the s pieces
- and for every permutation of the ℓ strands:

Store the number of possibilities to thread the ℓ strands from the bottom face to the rope.

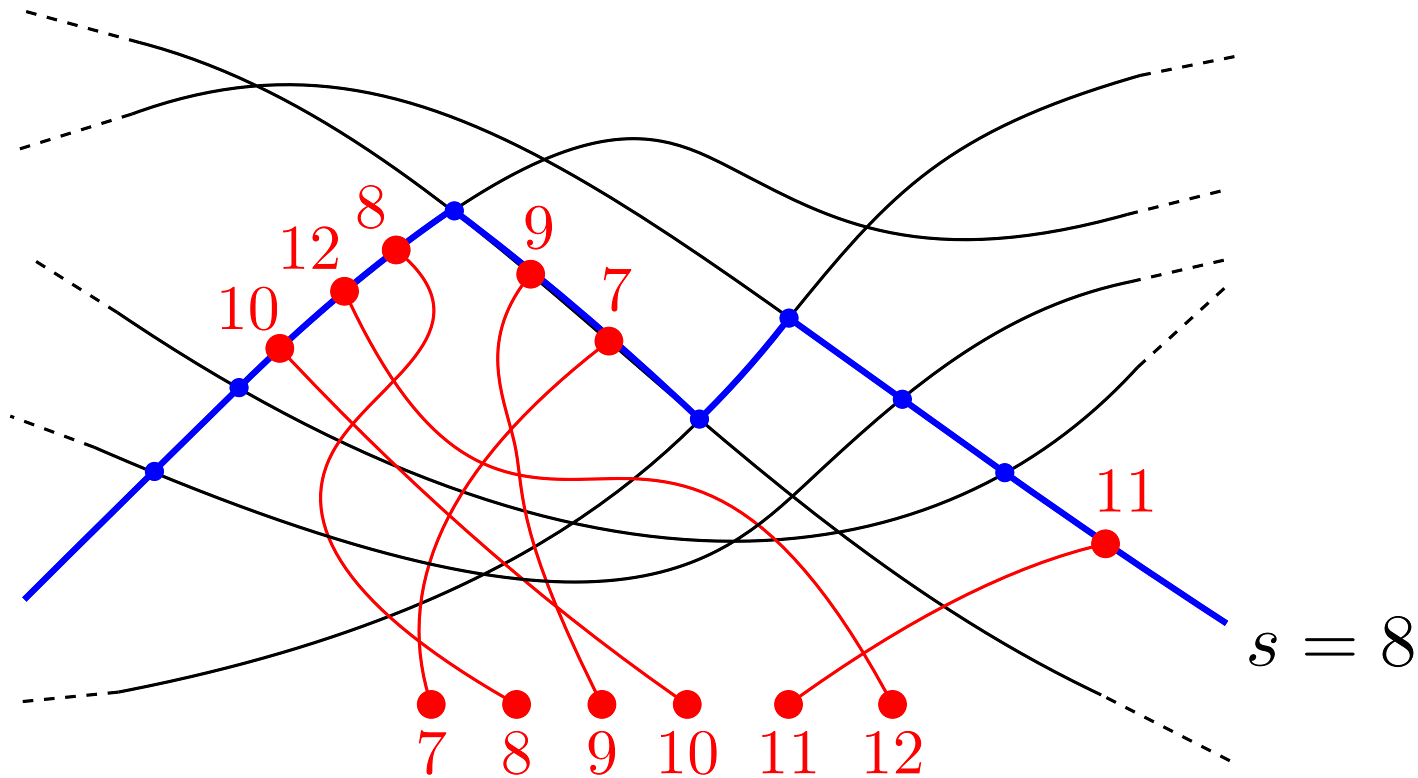
$$\to s(s+1)(s+2)\ldots(s+\ell-1) \text{ entries}$$

Dynamic programming

Freie Universität Berlin

For each rope:

(s pieces)

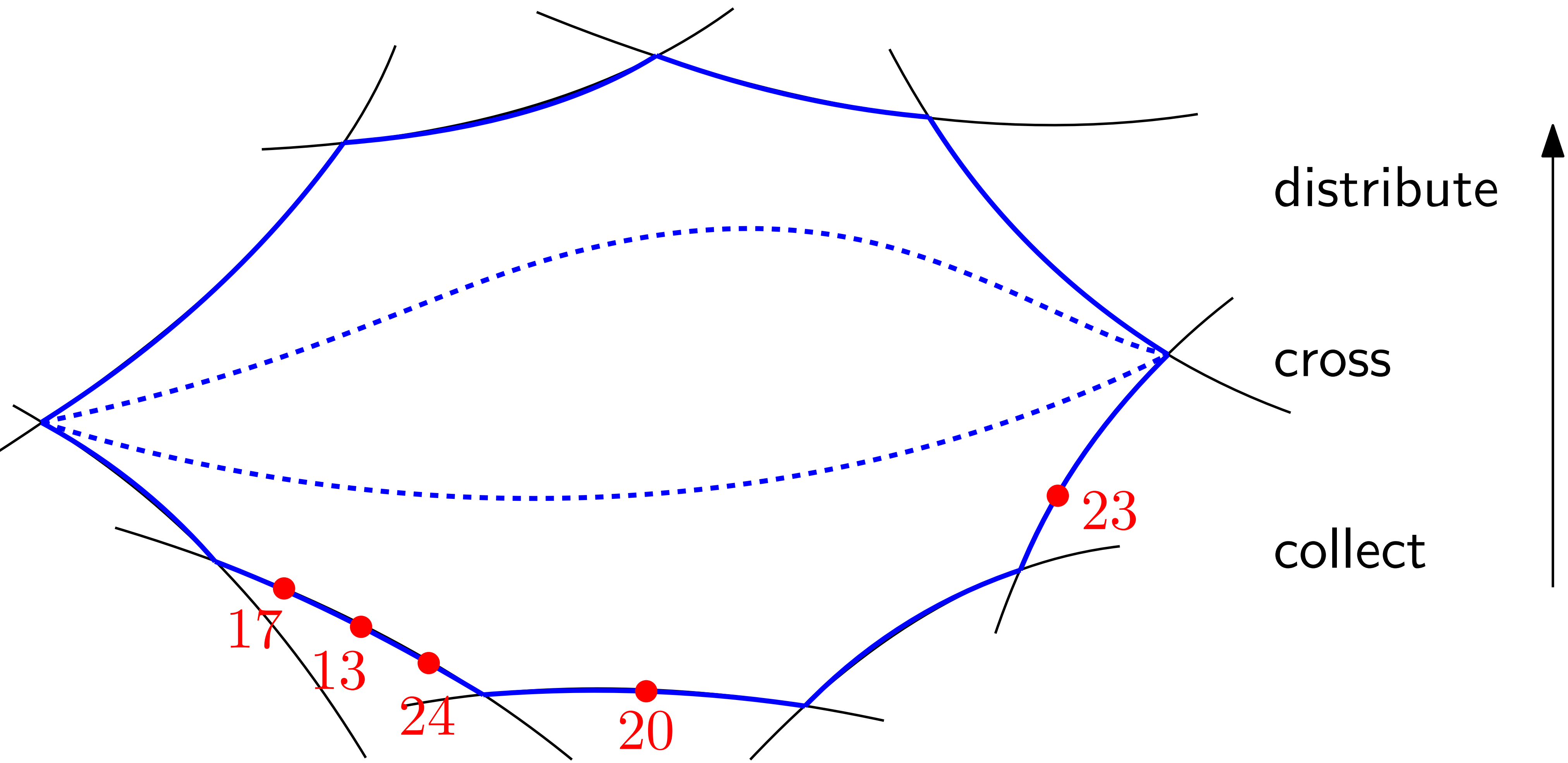


- For every distribution of the ℓ strands to the s pieces
- and for every permutation of the ℓ strands:

Store the number of possibilities to thread the ℓ strands from the bottom face to the rope.

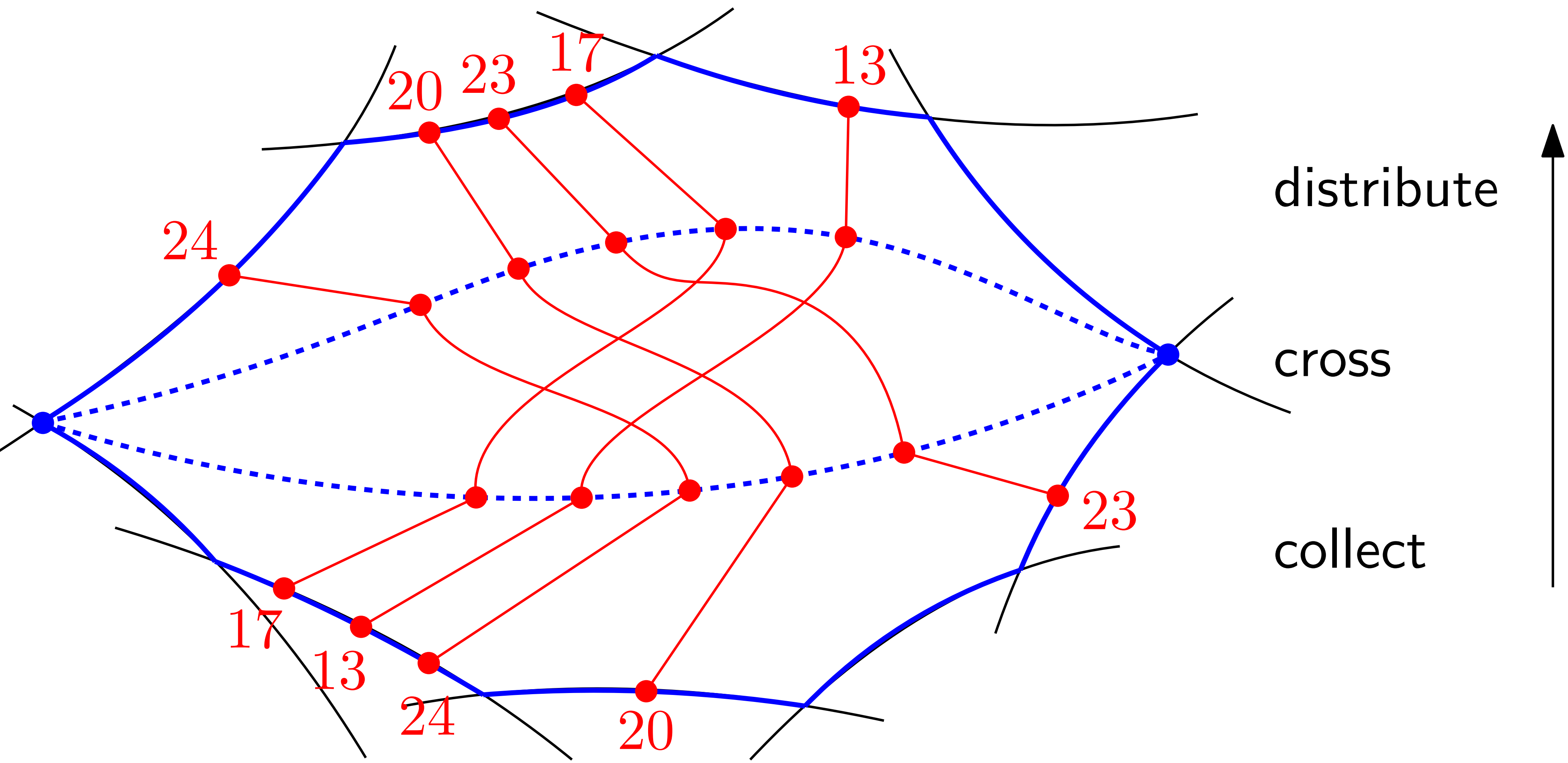
$\rightarrow s(s+1)(s+2)\dots(s+\ell-1)$ entries

Advancing the rope across a face



What is the contribution to the next rope?

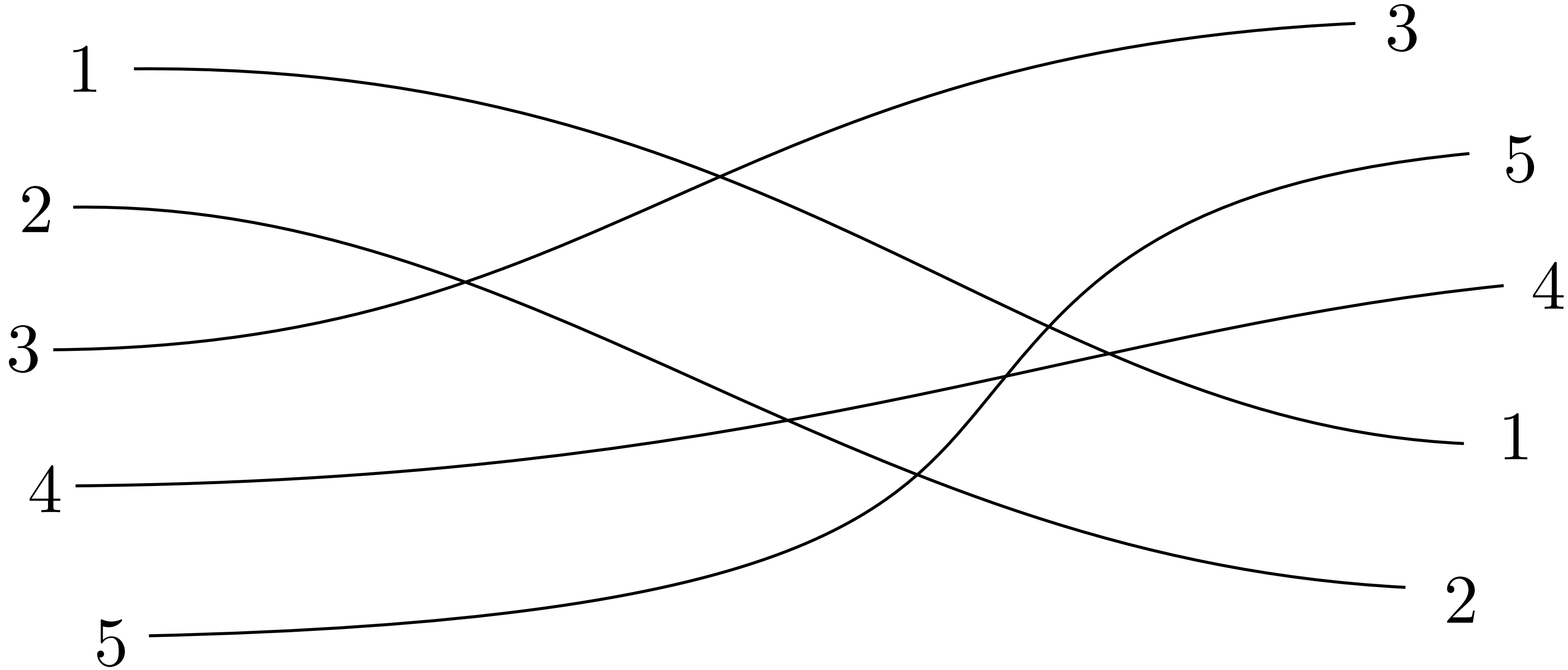
Advancing the rope across a face



What is the contribution to the next rope?

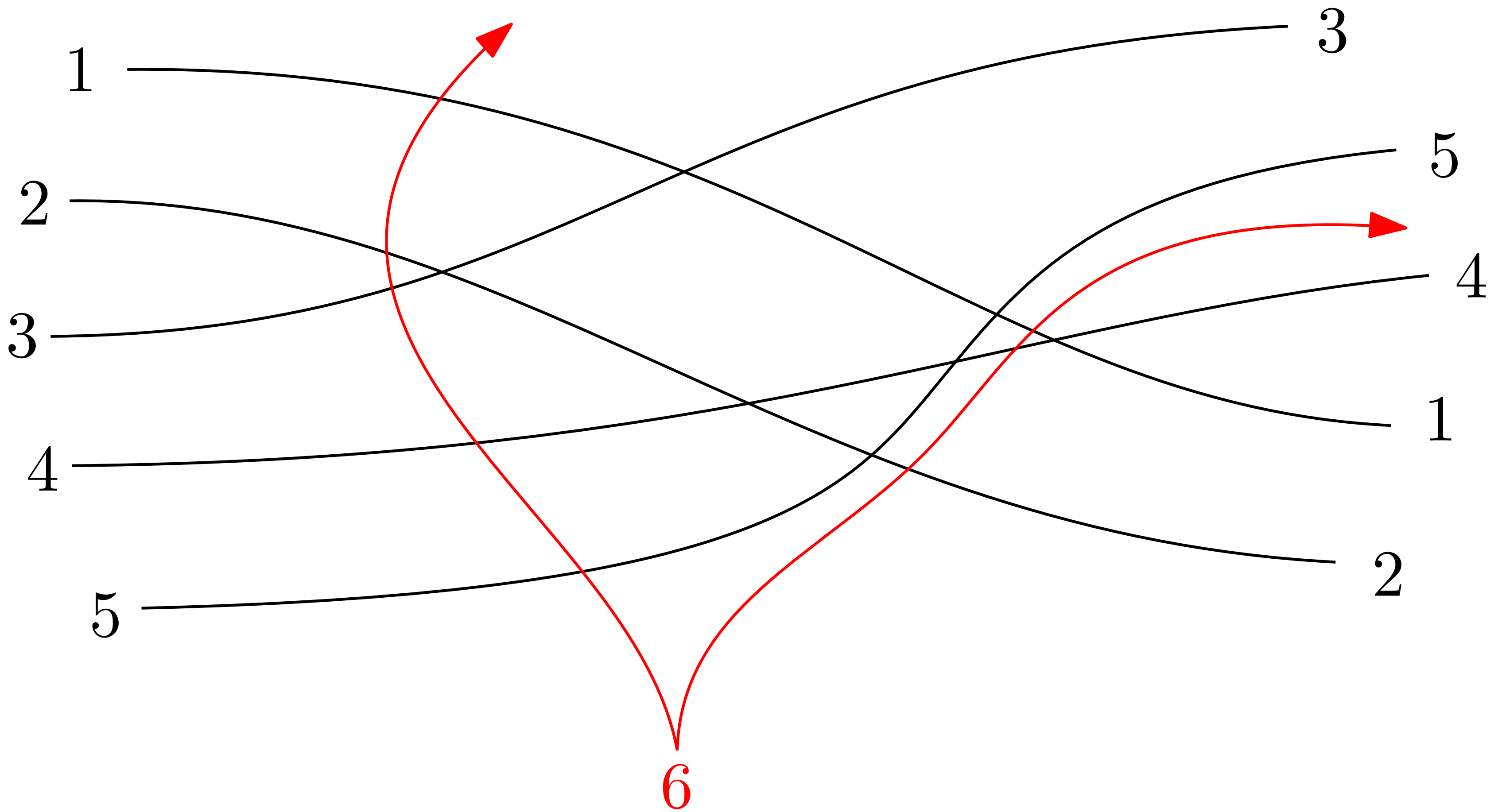
PARTIAL pseudoline arrangements

The ℓ pseudolines may cross or not.



PARTIAL pseudoline arrangements

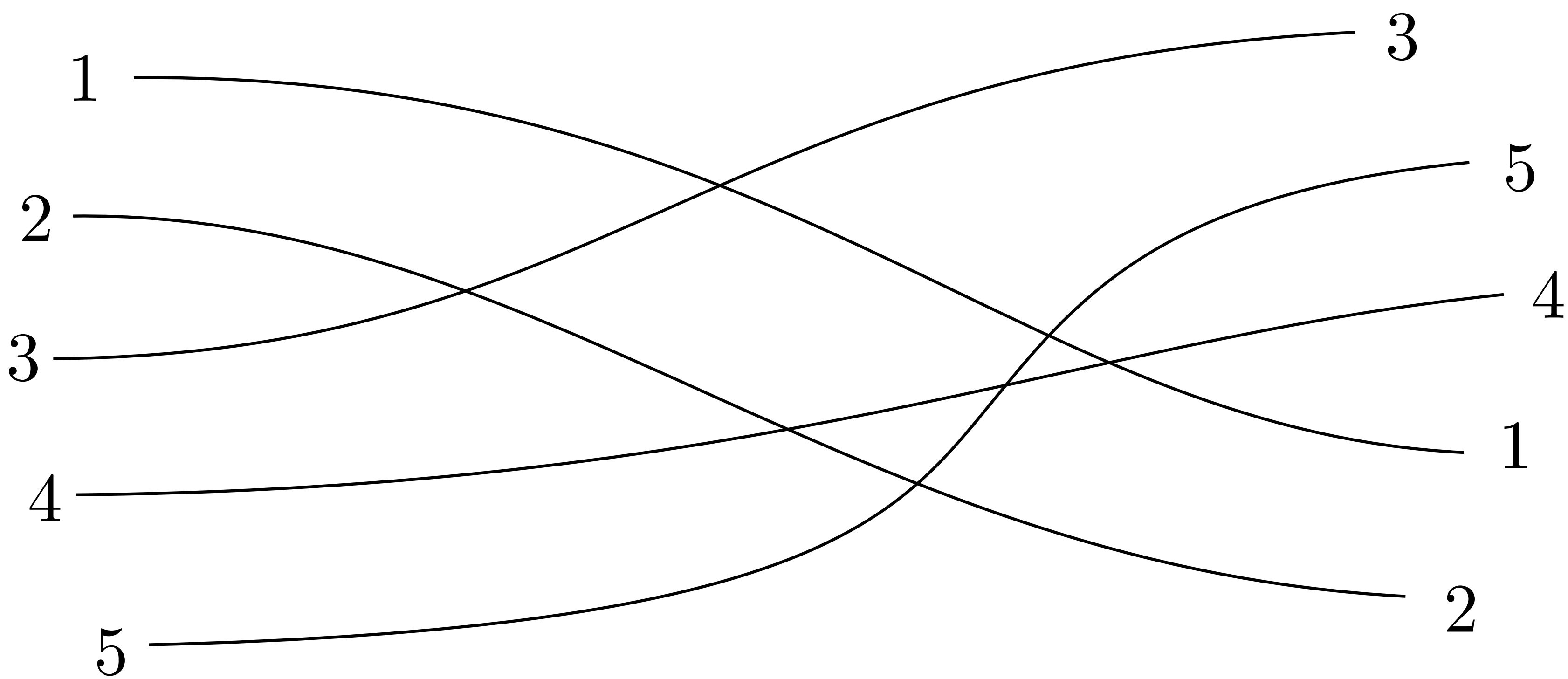
The ℓ pseudolines may cross or not.



Enumeration is as easy as for full PsA's.

PARTIAL pseudoline arrangements

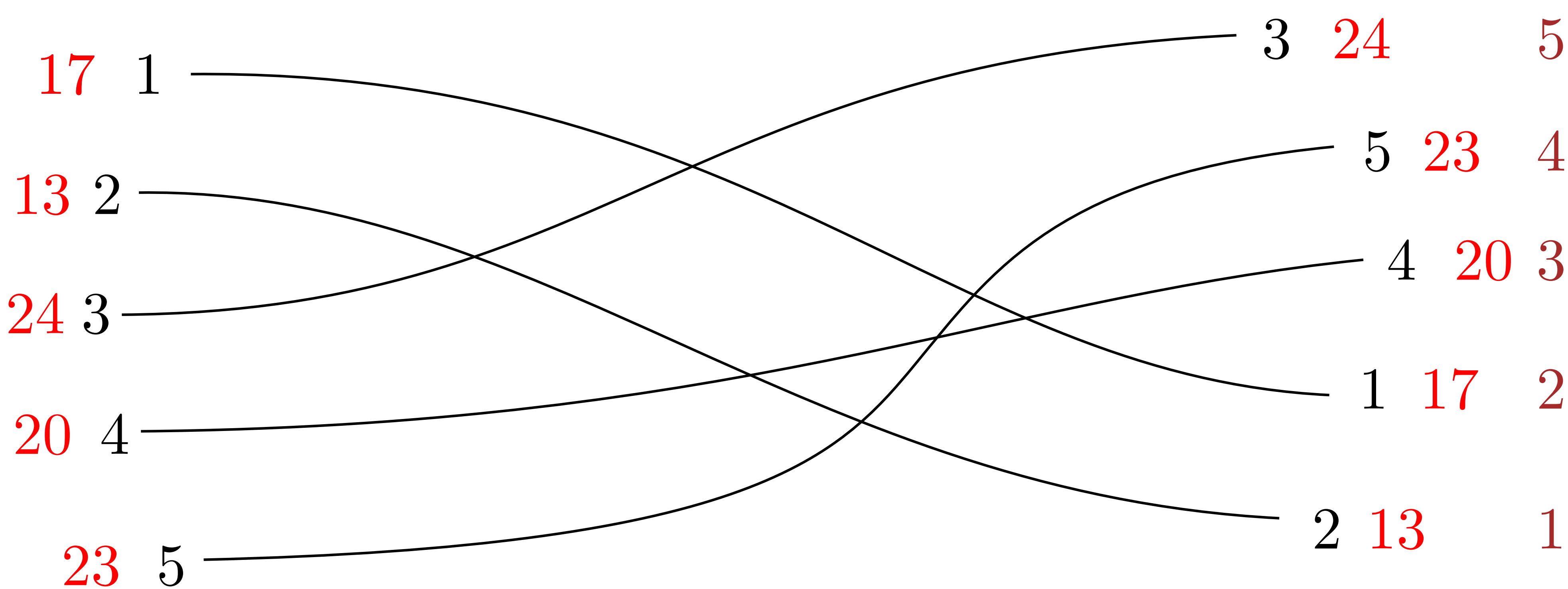
The ℓ pseudolines may cross or not.



Preprocessing: $\to$ $\ell!$ array

PARTIAL pseudoline arrangements

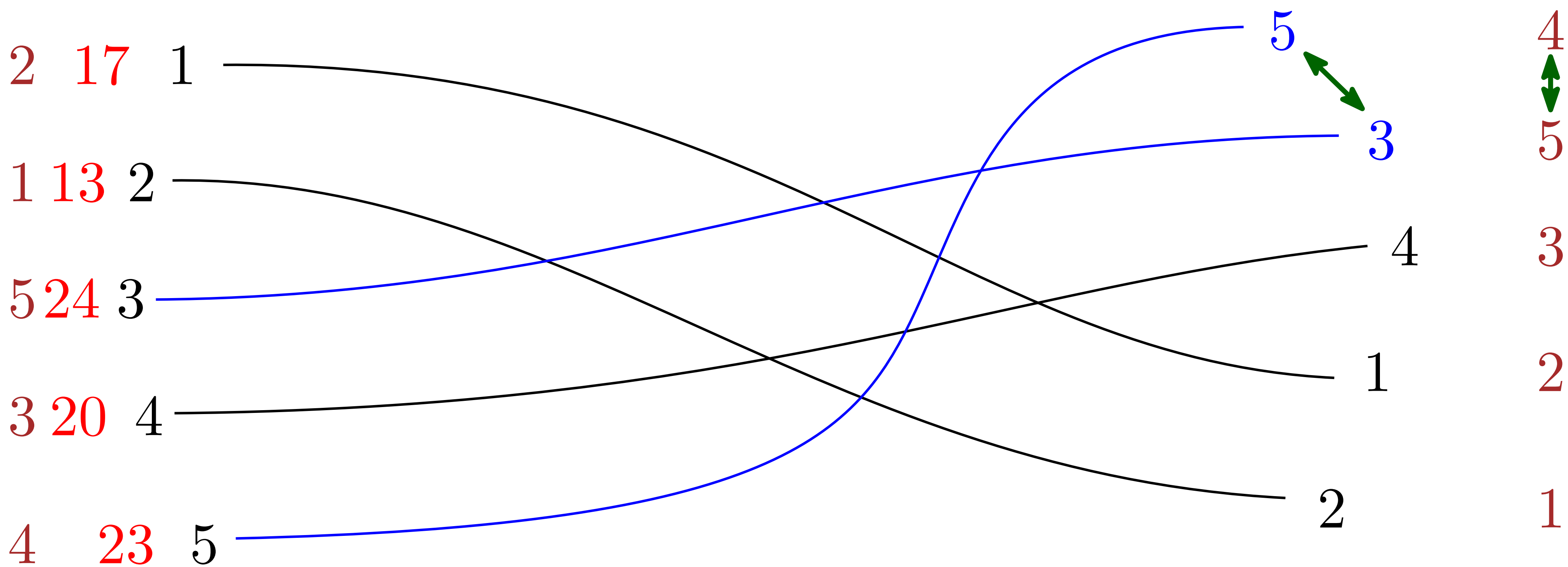
The ℓ pseudolines may cross or not.



Preprocessing: $\rightarrow$ $\ell!$ array

$\rightarrow$ $\ell! \times \ell!$ table (sparse!)

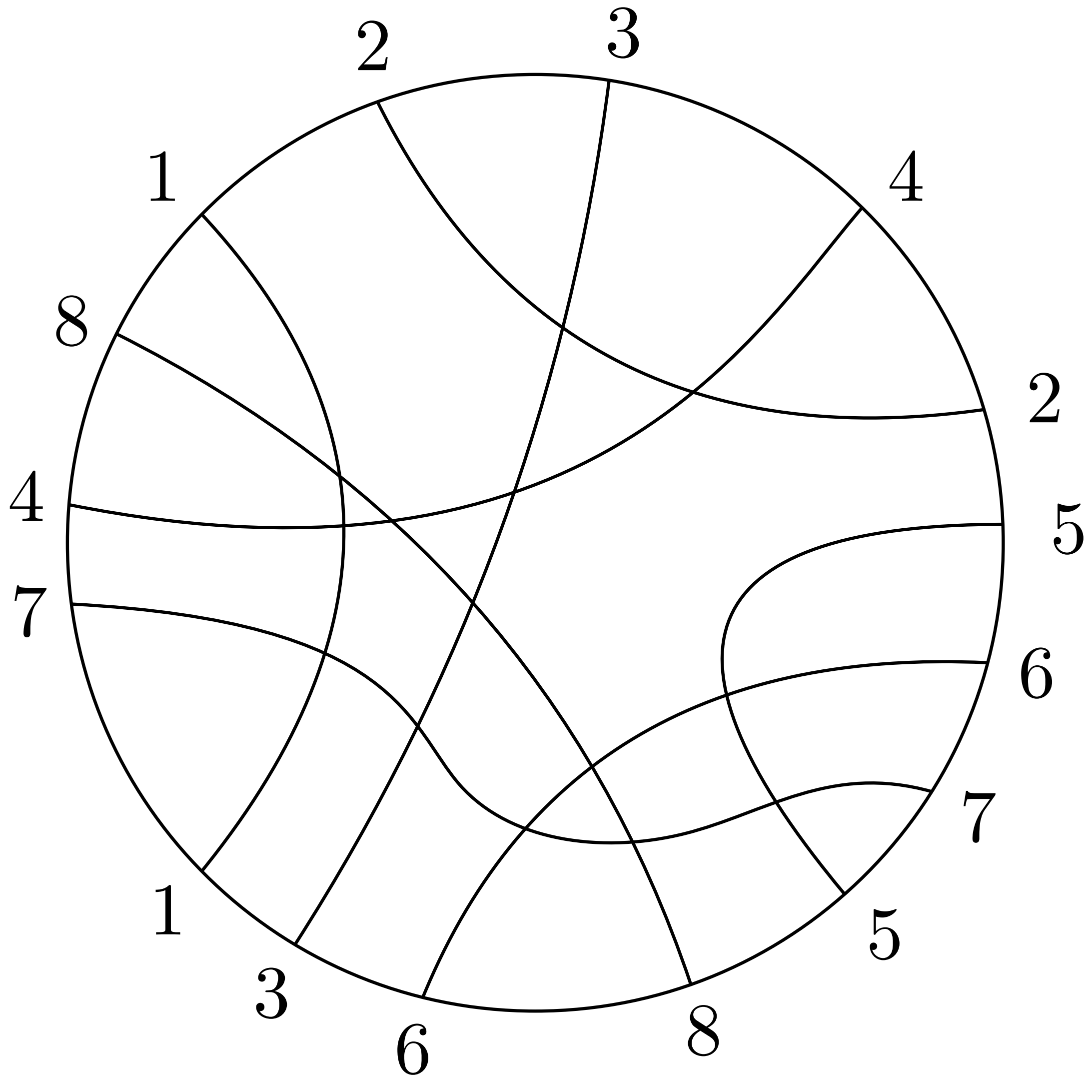
PARTIAL pseudoline arrangements



This is OK as a partial pseudoline arrangement, but not for the input sequence $17, 13, 27, 20, 23 = 2, 1, 5, 3, 4$.

PARTIAL pseudoline arrangements

Distinguish: more general *partial pseudoline arrangements* that are not necessarily x-monotone:

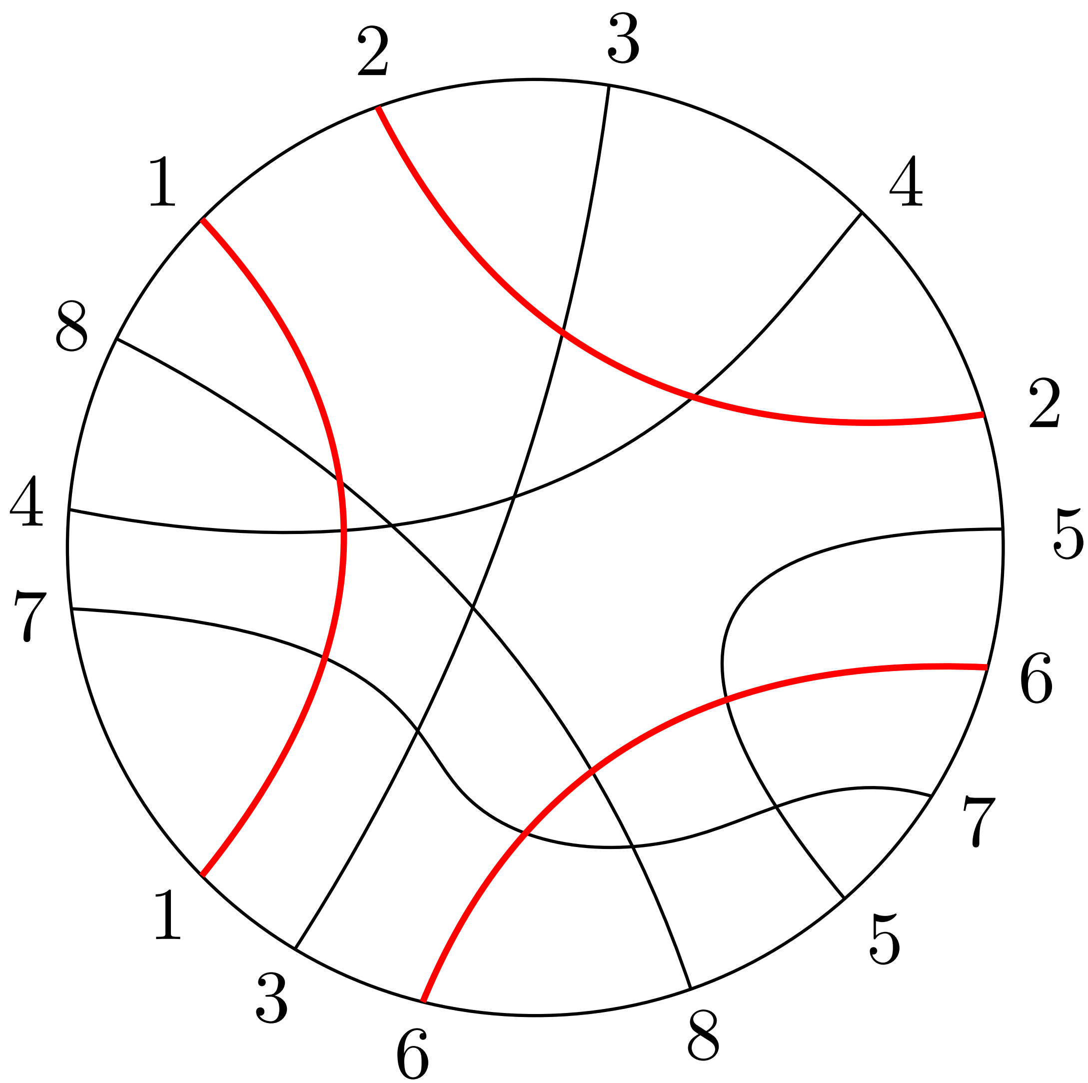


given by a *bipermutation* or (*matching*)

PARTIAL pseudoline arrangements

Freie Universität Berlin

Distinguish: more general *partial pseudoline arrangements* that are not necessarily x-monotone:



given by a *bipermutation* or (*matching*)

Algorithm summary

Freie Universität Berlin

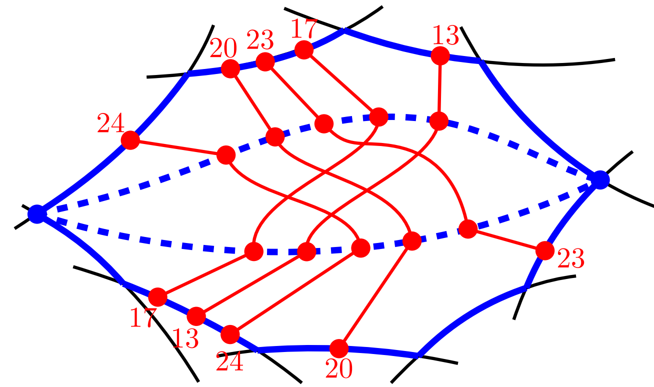
For each PsA of k pseudolines:

- Compute a sweep by ropes
- For each rope:
  - For each distribution and permutation of the ℓ strands:
    - Compute the contributions to the next rope, and accumulate them.

Network model

PSA $\equiv$ source-to-sink path

levels $\hat{\approx}$ ropes



distribute

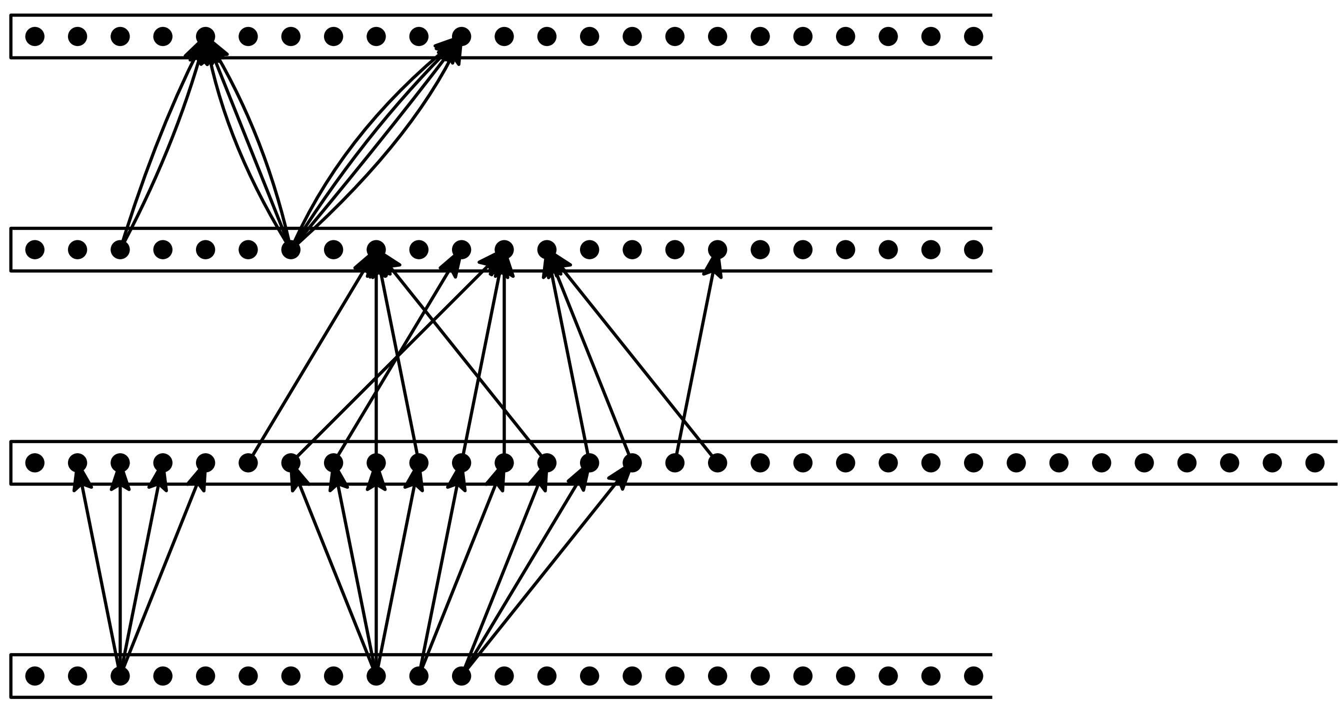
cross

collect

cross

collect

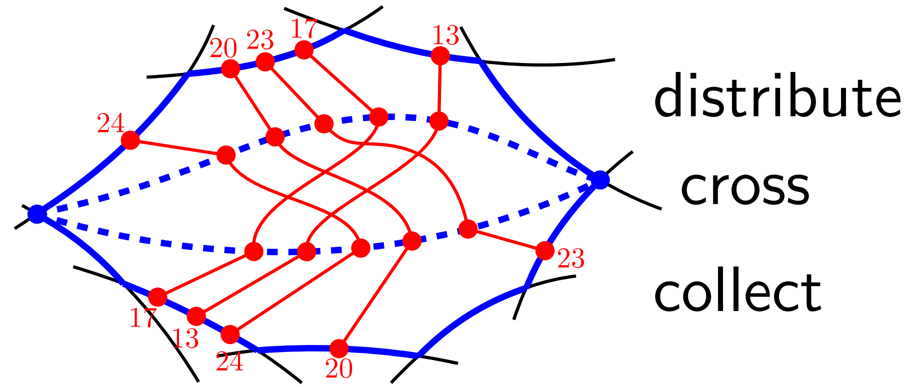
distribute

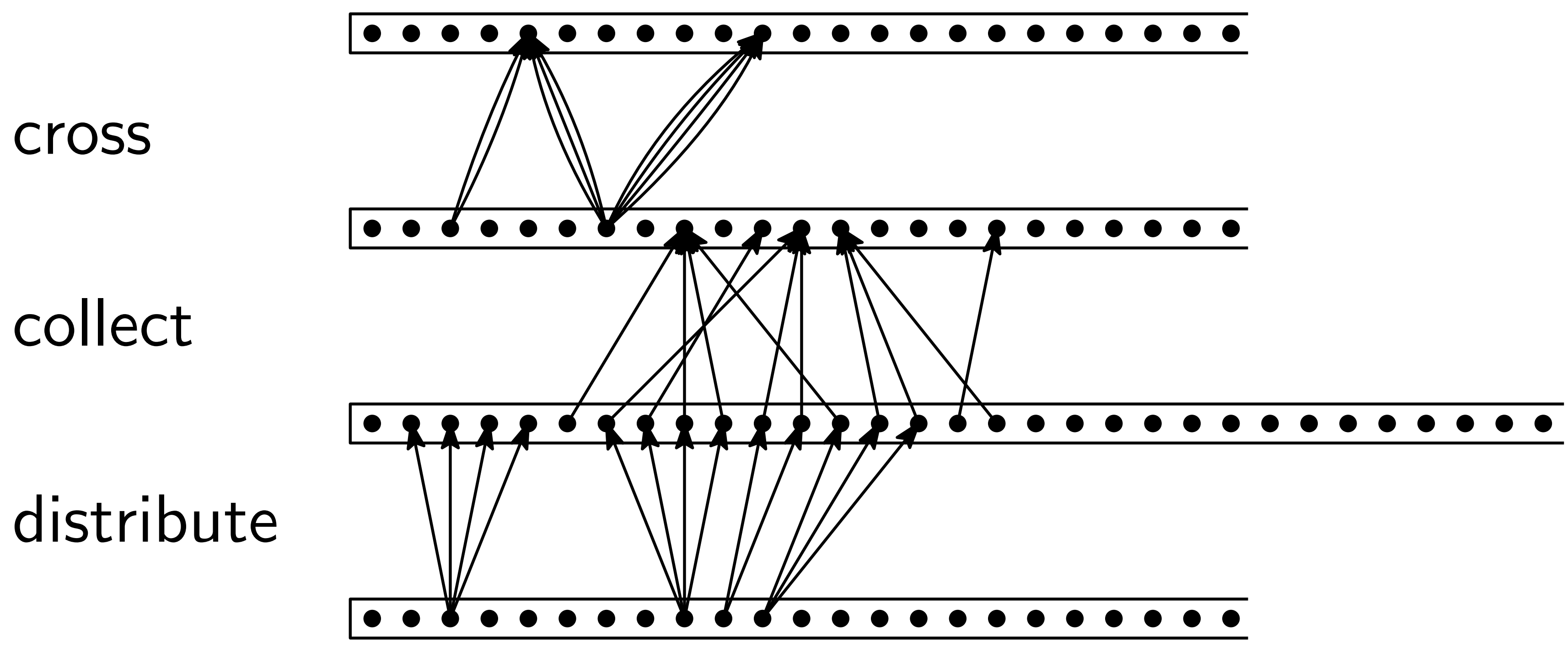


Network model

PSA $\equiv$ source-to-sink path

levels $\hat{\approx}$ ropes

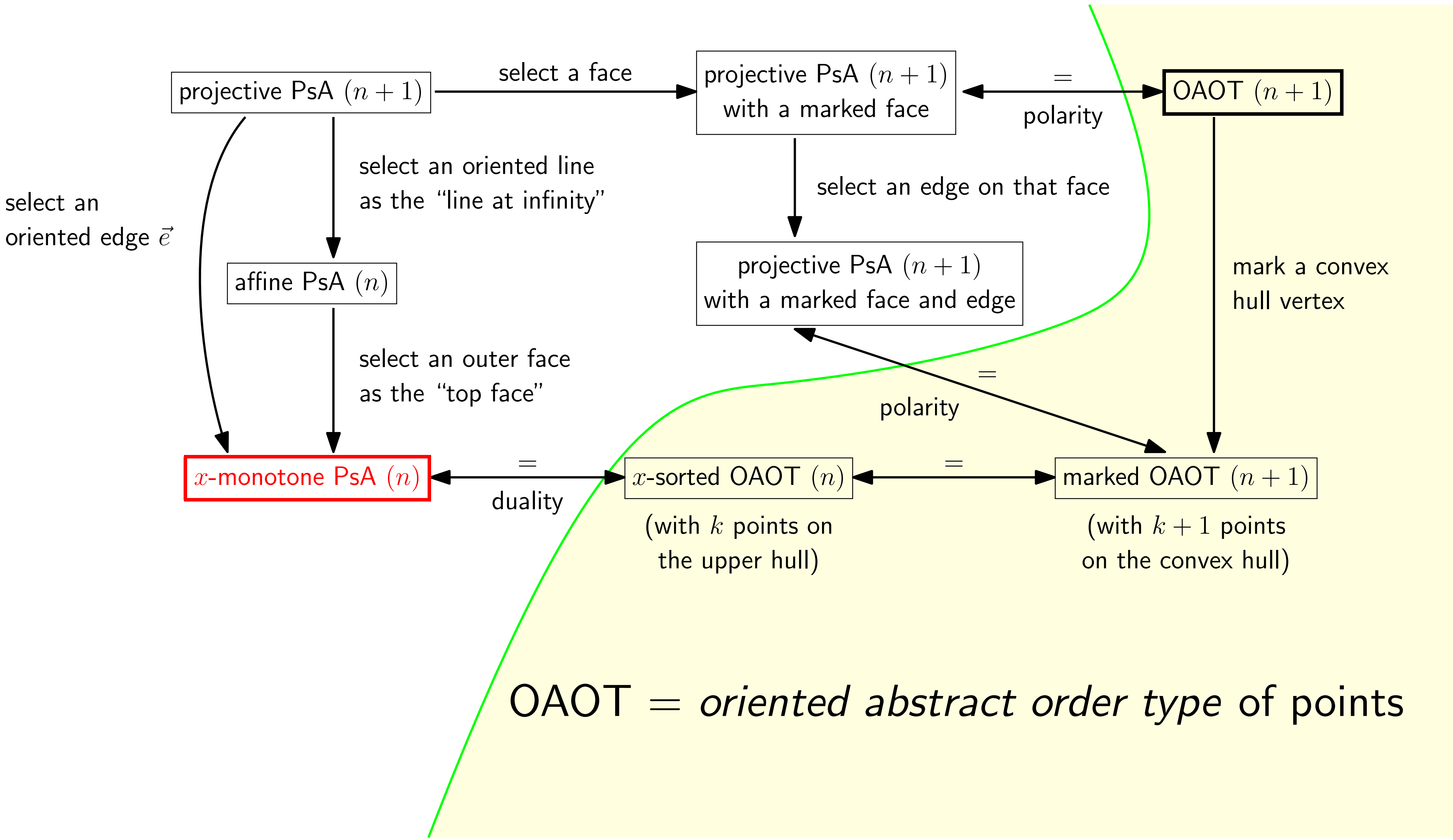




Some implementation details

- PYTHON, with `scipy` for large arrays of 32/64-bit integers
- modular arithmetic with 3 moduli: 2^{64}, two 30-bit numbers
- $n = 16 = k + \ell = 7 + 9$. Large memory! Max. "rope" $s = 7$. 256 GBytes is enough; 128 GBytes sometimes failed.
- easy to parallelize: 24,698 independent tasks
- total CPU time: about 5.5 months, using various workstations of different speeds
- CPU time for $n = 15 = 6 + 9$ (exploiting symmetry): 6 h. By contrast: PYTHON without `scipy` took 50 CPU days. (using a greedy rope)
- There is also a version in C (using `CWEB`) for the task of *enumerating* PsA's $\rightarrow$ OAOT of 13 points [OEIS A006247]

What else to enumerate



OAOT = *oriented abstract order type* of points

The required rope length

- Every arrangement requires $\geq n+1$ pieces (for $n \geq 3$).
- can always do with $\leq 2n-2$ pieces. (greedy sweep)
- Some arrangements require $\lfloor \frac{7n}{4} \rfloor - 1$ pieces.

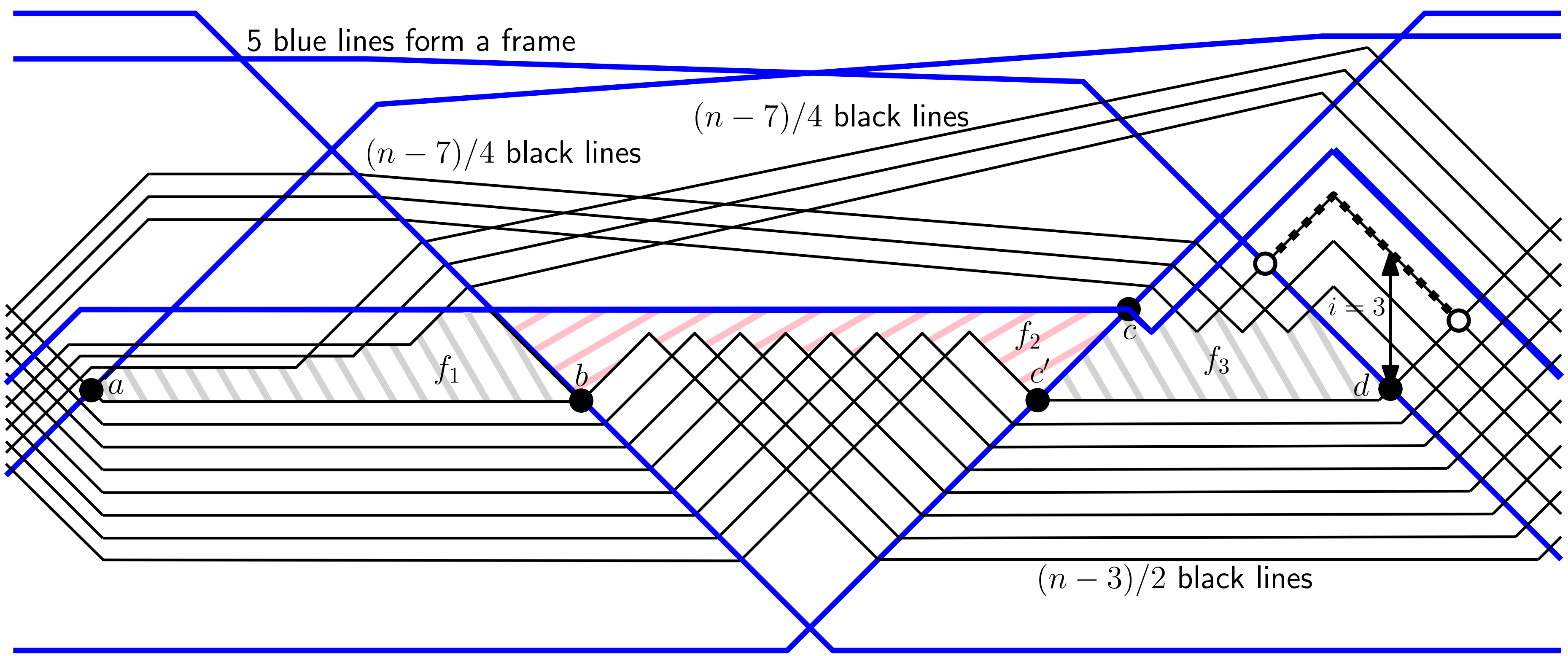
  (This is the true maximum for $n \leq 9$.)
- NP-hard? (homotopy height, cutwidth)

  [Biedl, Chambers, Kostitsyna, Rote, 2020/2022/2024?, unpublished]

The required rope length

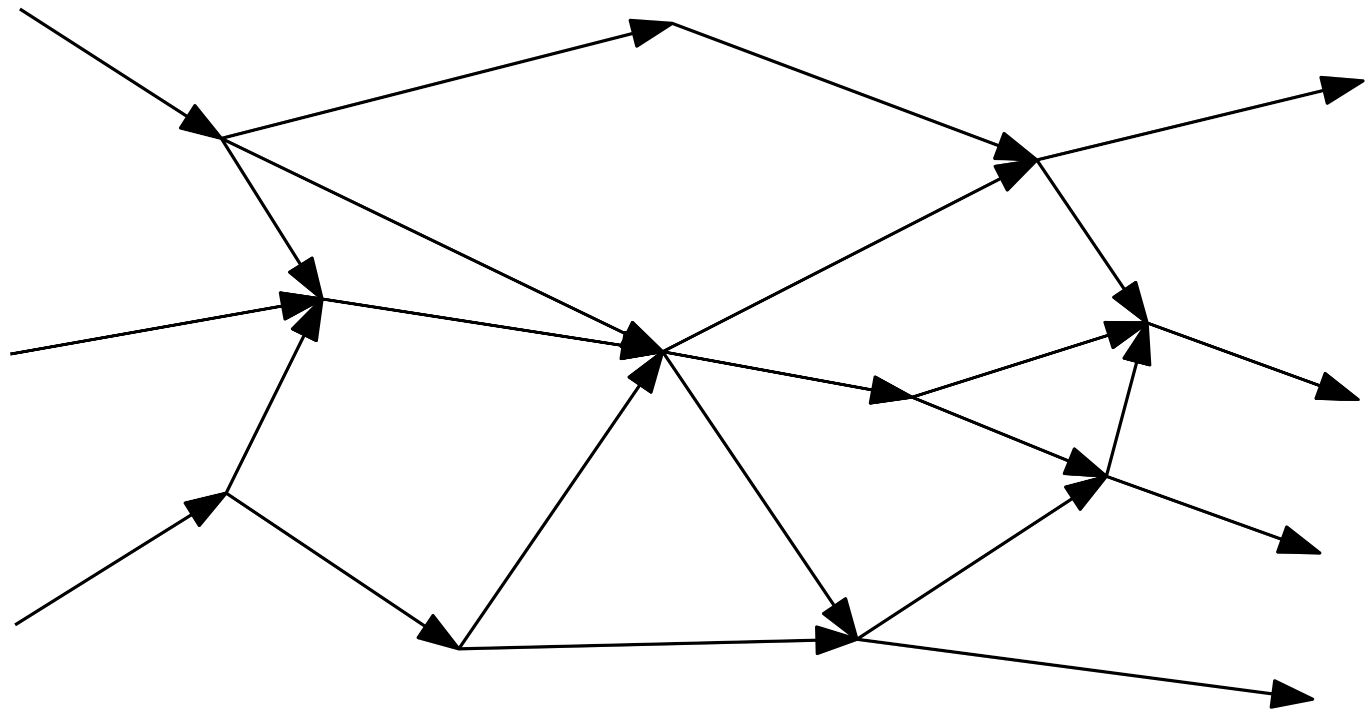
- Every arrangement requires $\geq n+1$ pieces (for $n \geq 3$).
- can always do with $\leq 2n-2$ pieces. (greedy sweep)
- Some arrangements require $\lfloor \frac{7n}{4} \rfloor - 1$ pieces.
  (This is the true maximum for $n \leq 9$.)
- NP-hard? (homotopy height, cutwidth)

[Biedl, Chambers, Kostitsyna, Rote, 2020/2022/2024?, unpublished]



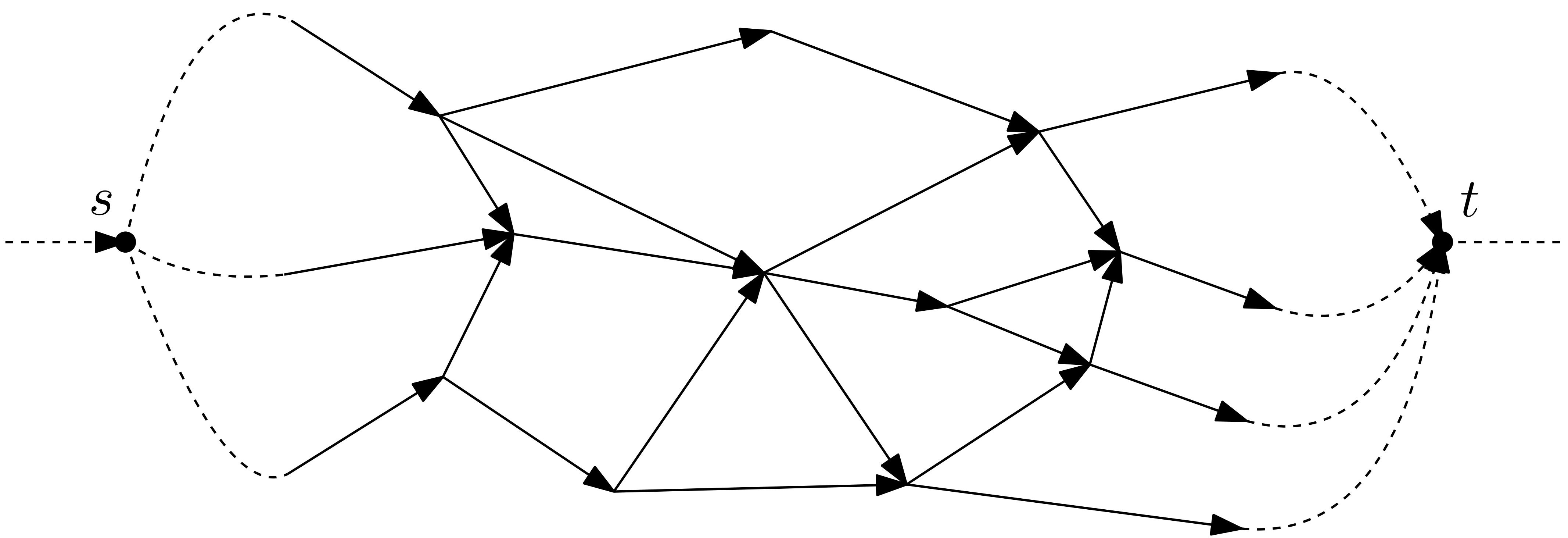
The required rope length

This is really about *bipolar orientations* (s-t-planar DAGs):



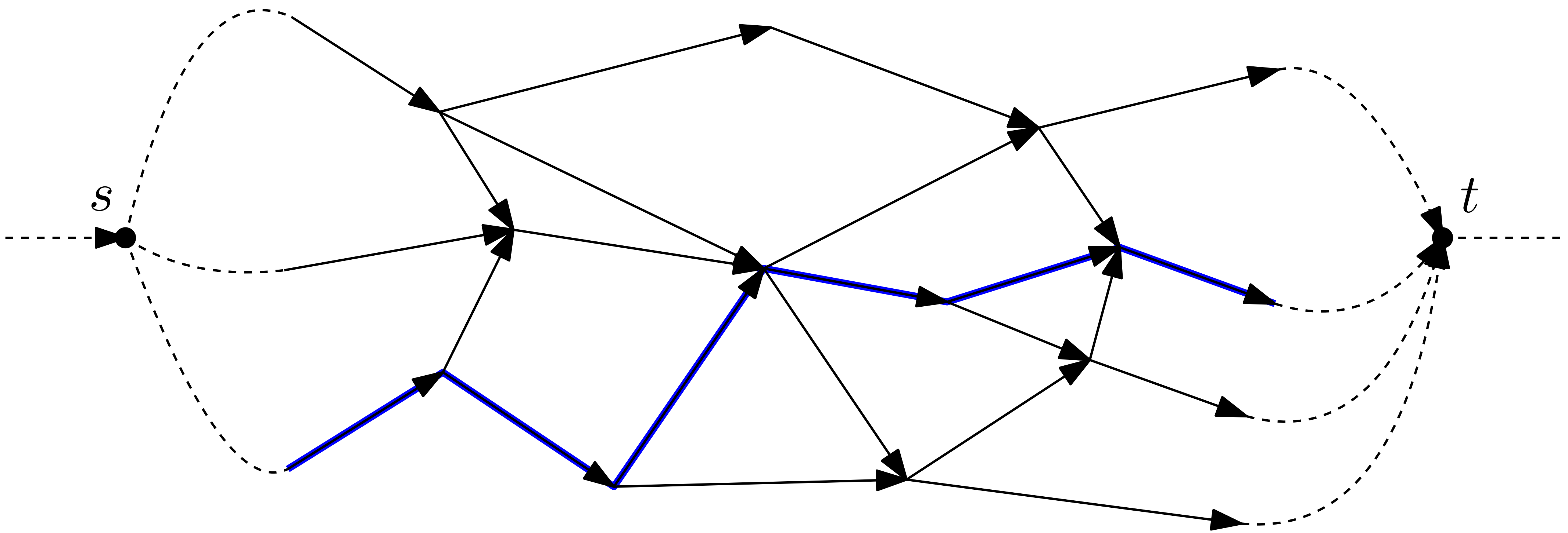
The required rope length

This is really about *bipolar orientations* (s-t-planar DAGs):



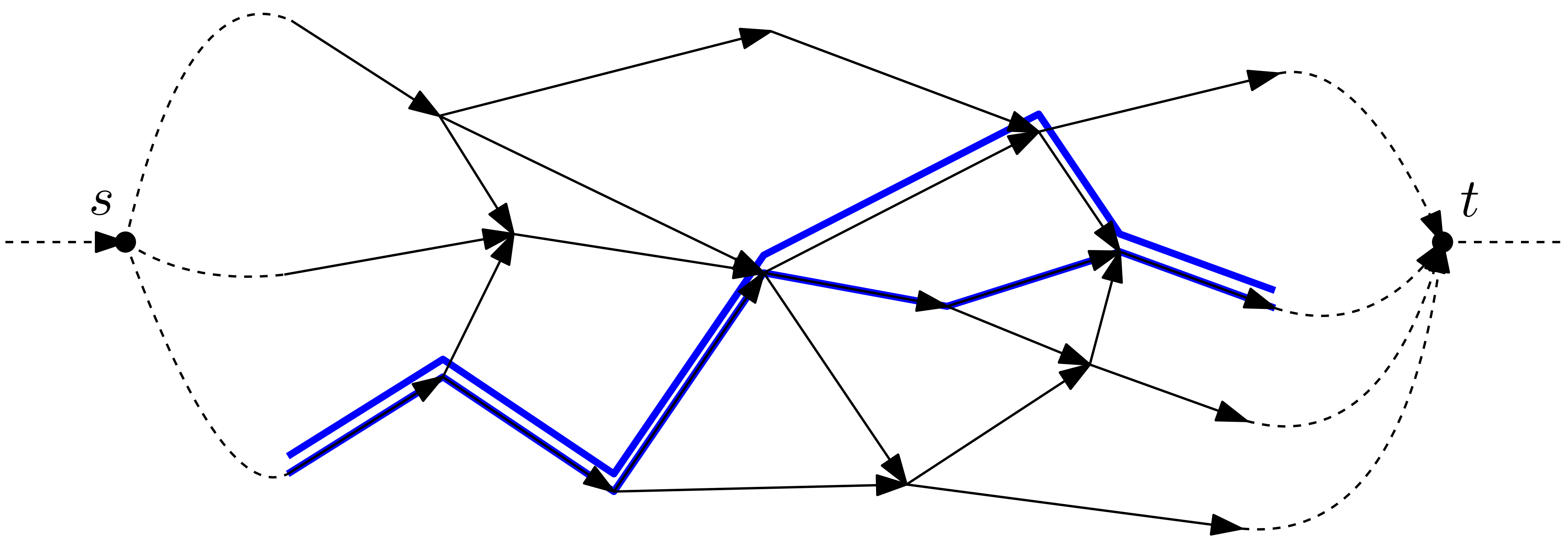
The required rope length

This is really about *bipolar orientations* (s-t-planar DAGs):



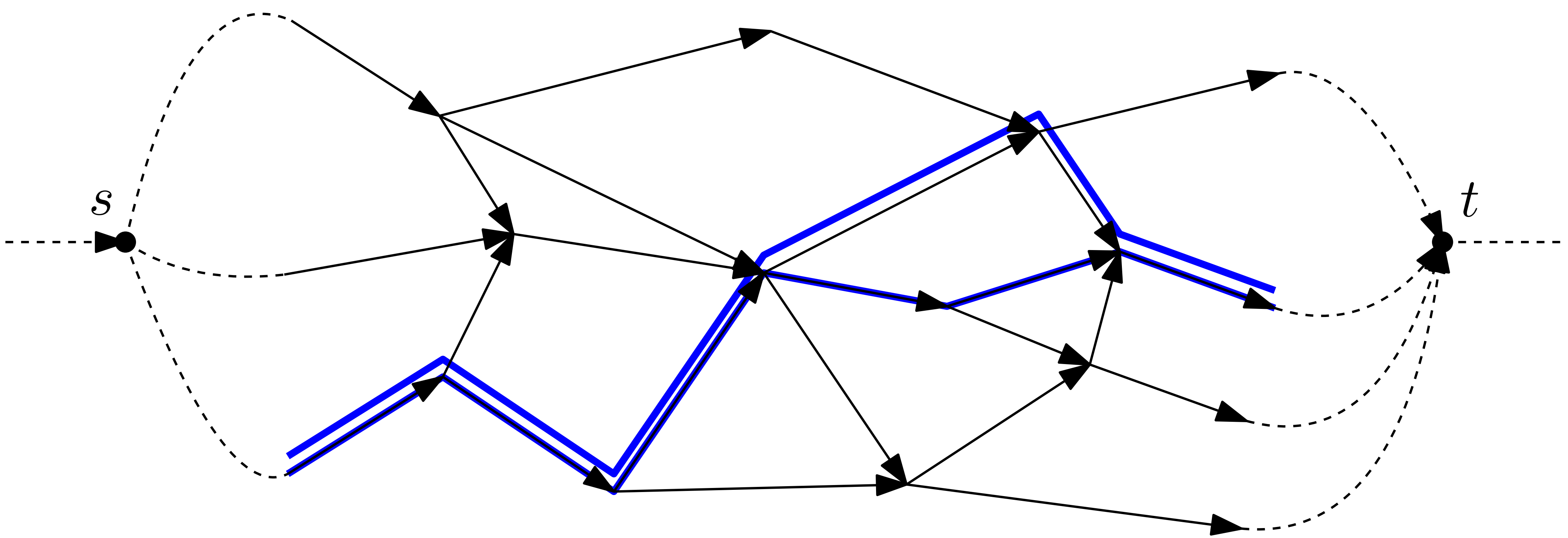
The required rope length

This is really about *bipolar orientations* (s-t-planar DAGs):



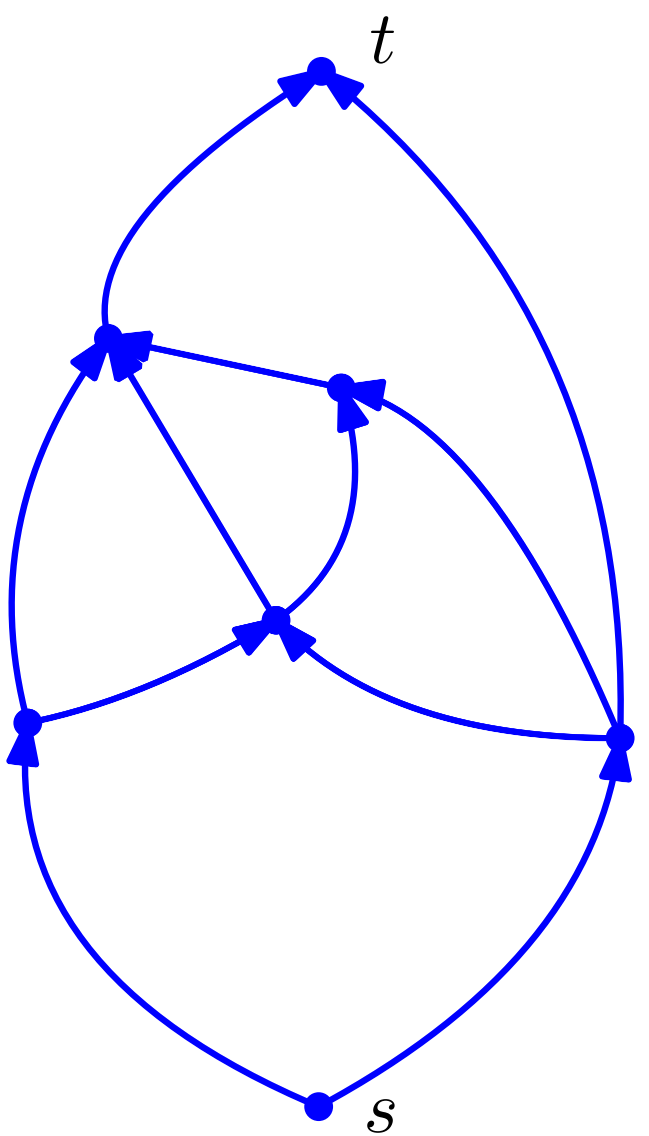
The required rope length

This is really about *bipolar orientations* (s-t-planar DAGs):



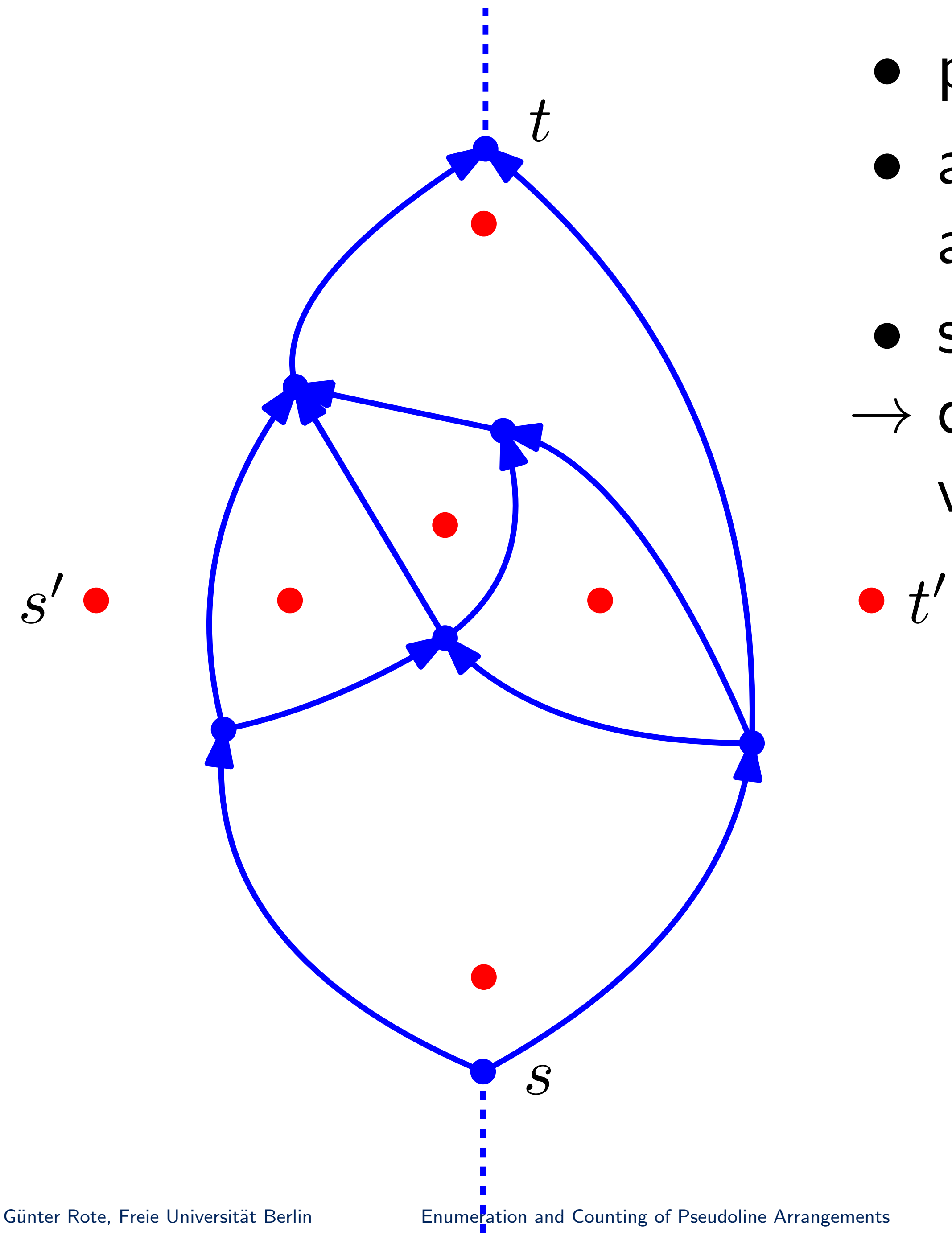
“leftmost-first” greedy sweep
$\rightarrow$ coordinated simultaneous primal-dual sweep

Bipolar orientations (s-t-planar graphs)



- plane directed acyclic graph
- a single source s and
  a single sink t

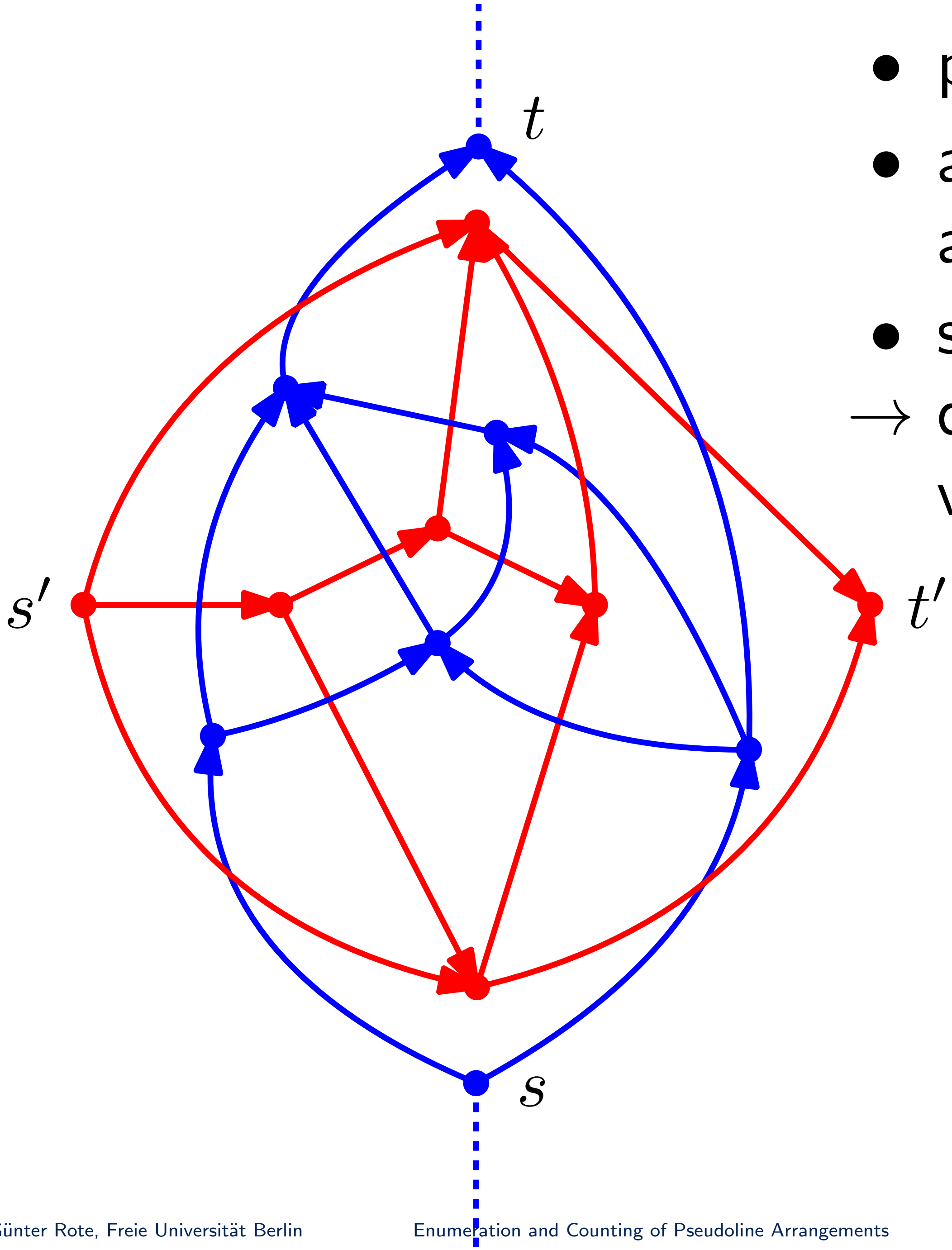
Bipolar orientations (s-t-planar graphs)



- plane directed acyclic graph
- a single source s and a single sink t
- split the outer face:

$\rightarrow$ dual graph with a *left* outer vertex s' and a *right* vertex t'

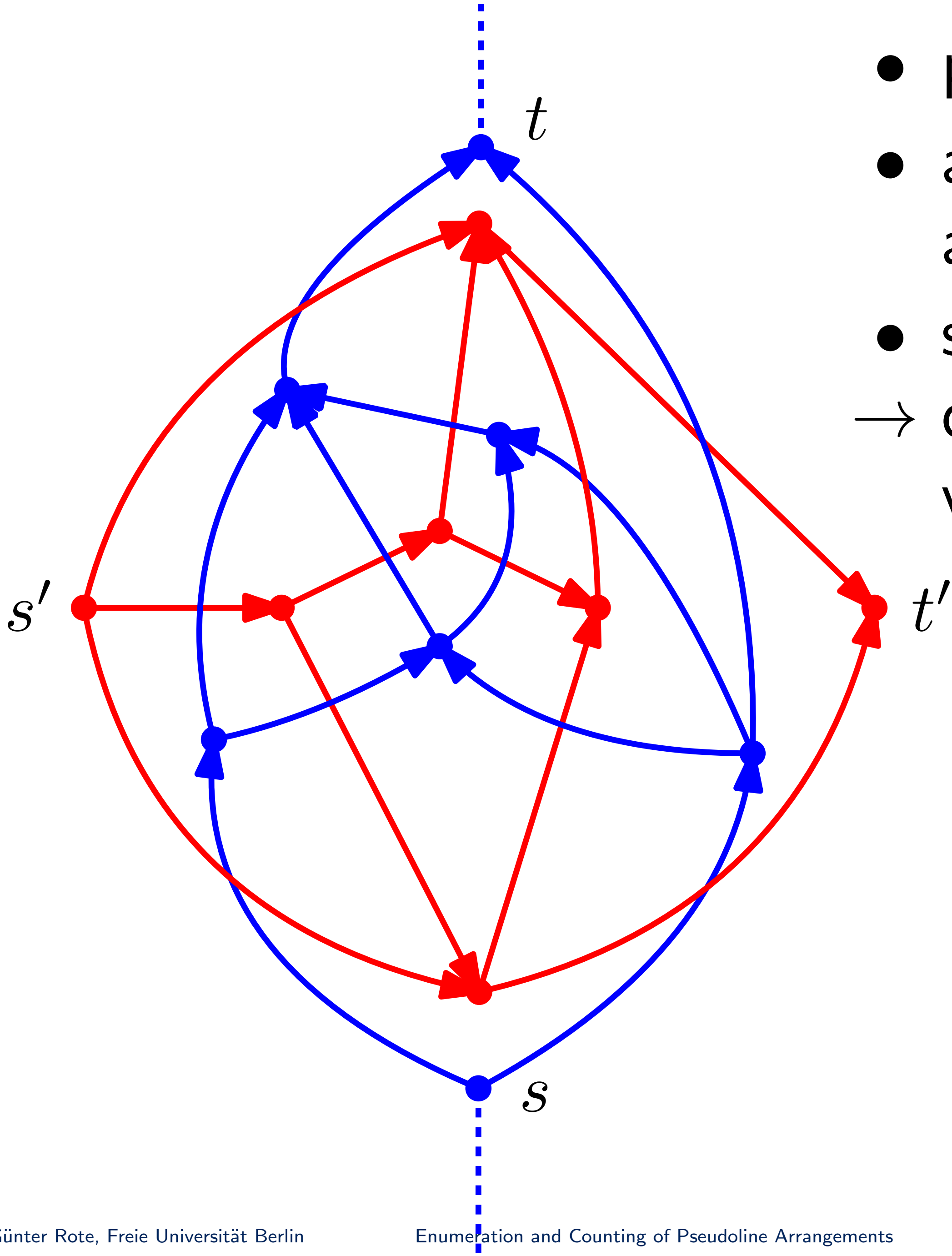
Bipolar orientations (s-t-planar graphs)



- plane directed acyclic graph
- a single source s and a single sink t
- split the outer face:

$\rightarrow$ dual graph with a *left* outer vertex s' and a *right* vertex t'

Bipolar orientations (s-t-planar graphs)



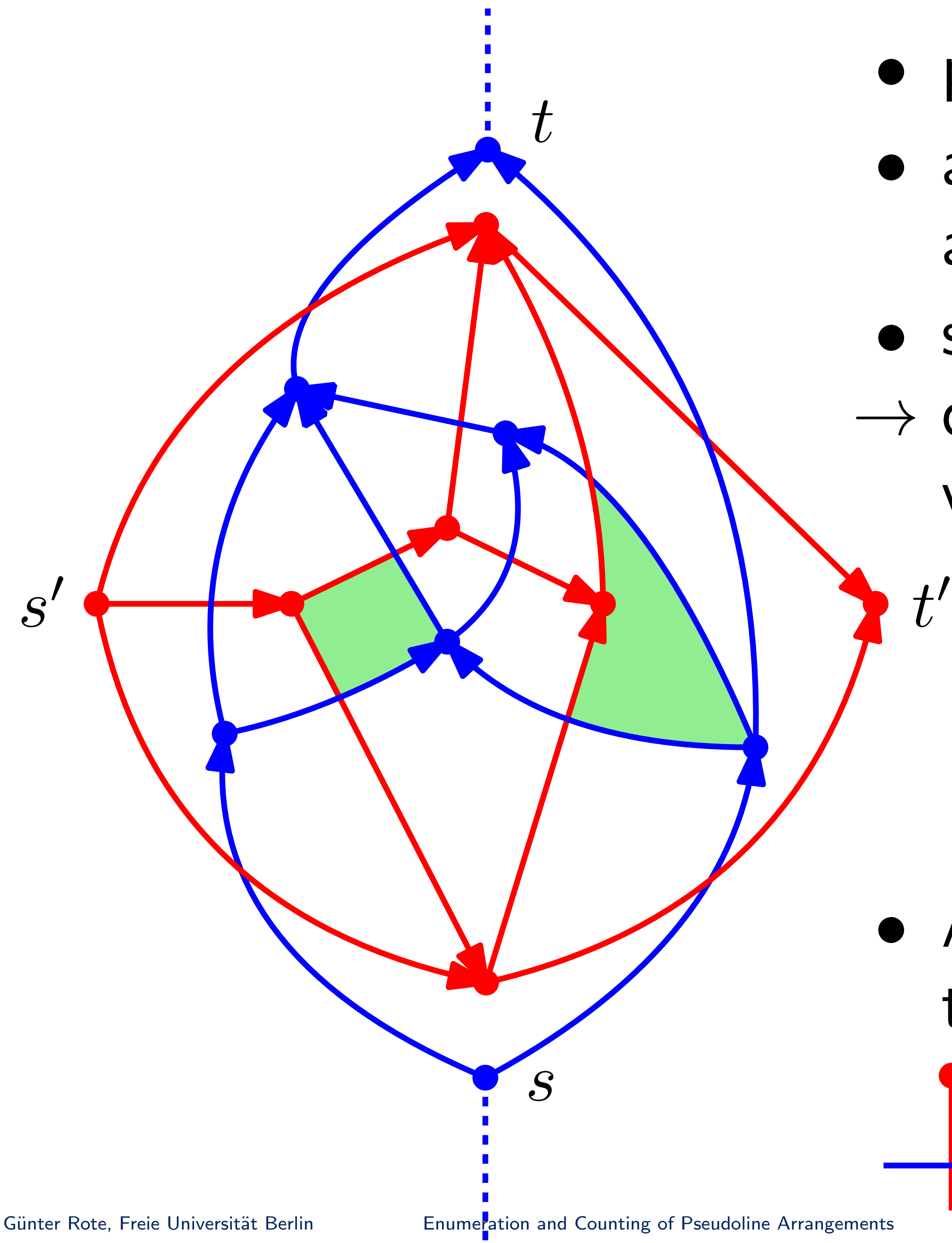
- plane directed acyclic graph
- a single source s and a single sink t
- split the outer face:

$\rightarrow$ dual graph with a *left* outer vertex s' and a *right* vertex t'

- The dual graph is also a bipolar orientation. (may be a multigraph)

Bipolar orientations (s-t-planar graphs)

Freie Universität Berlin

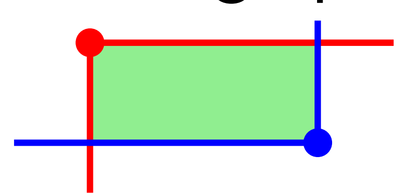


- plane directed acyclic graph
- a single source s and a single sink t
- split the outer face:

$\rightarrow$ dual graph with a *left* outer vertex s' and a *right* vertex t'

  - The dual graph is also a bipolar orientation. (may be a multigraph)

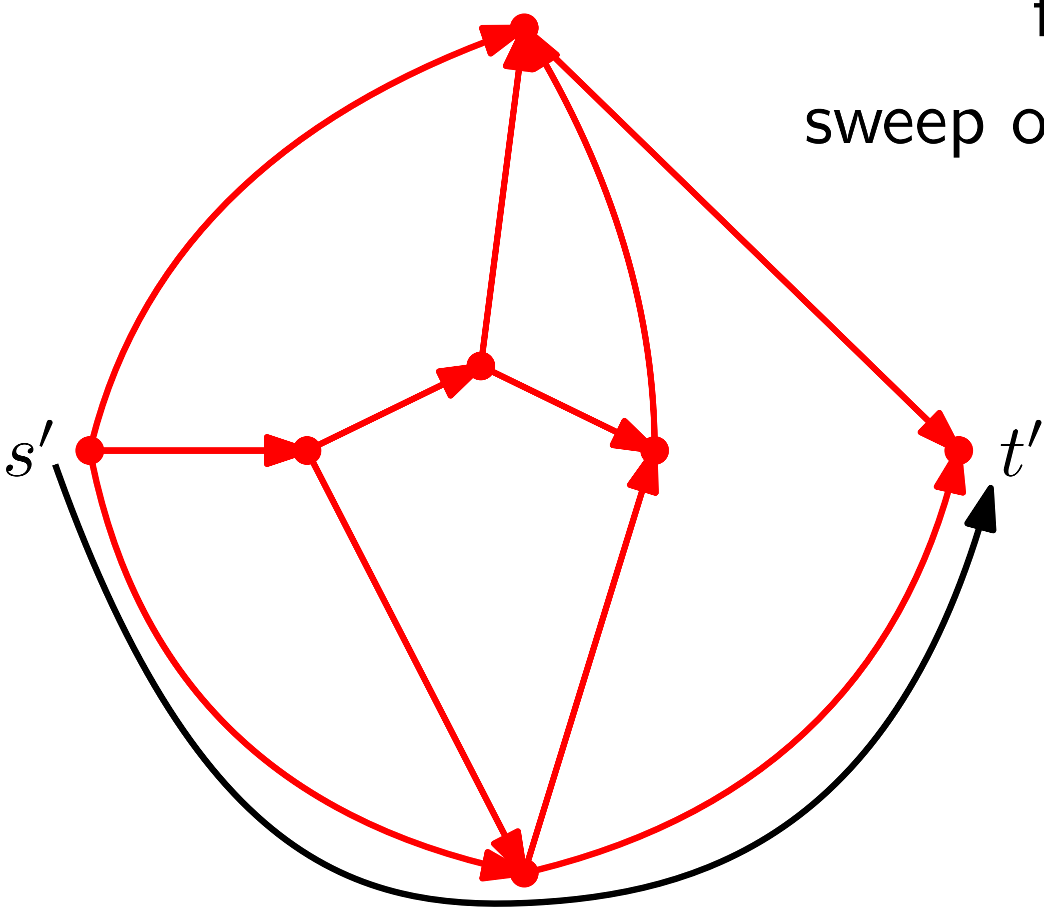
- All faces in the overlay of the two graphs are quadrilaterals:



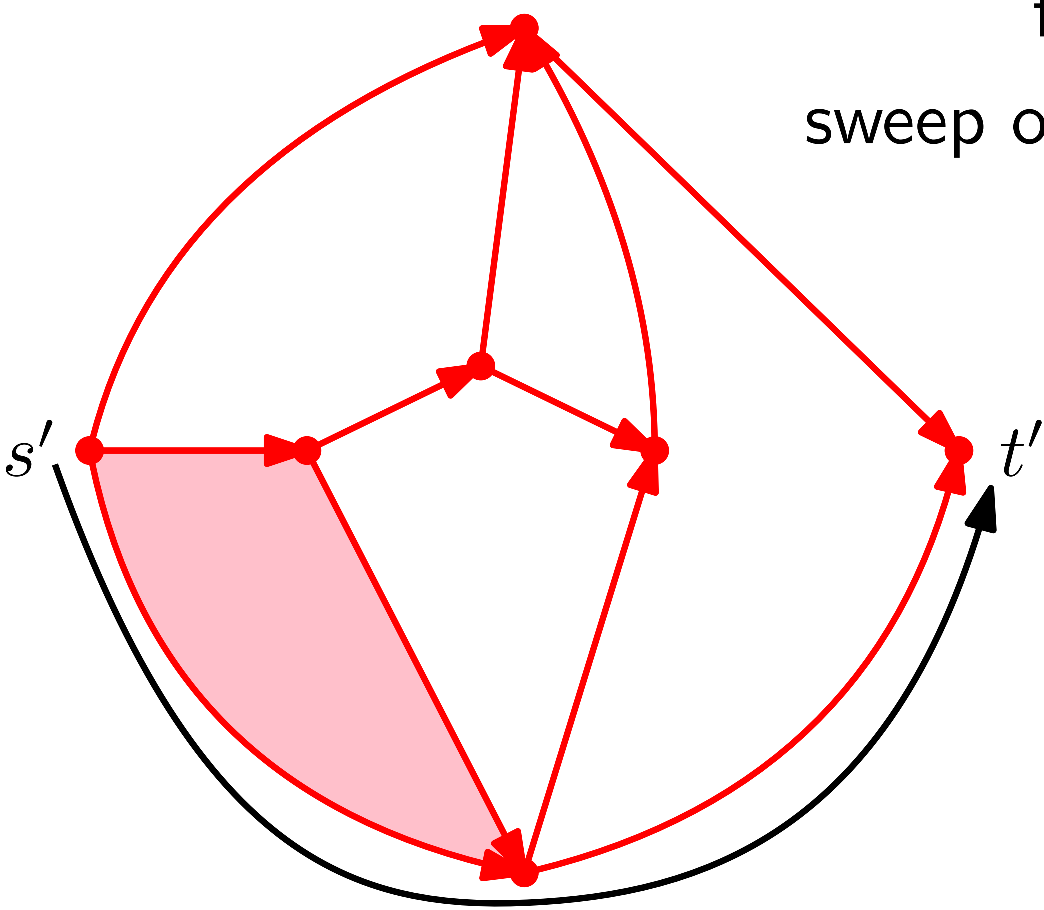
Coordinated primal-dual sweep

- sweep the dual graph with an s'–t' rope from bottom to top

sweep over the *leftmost* possible face



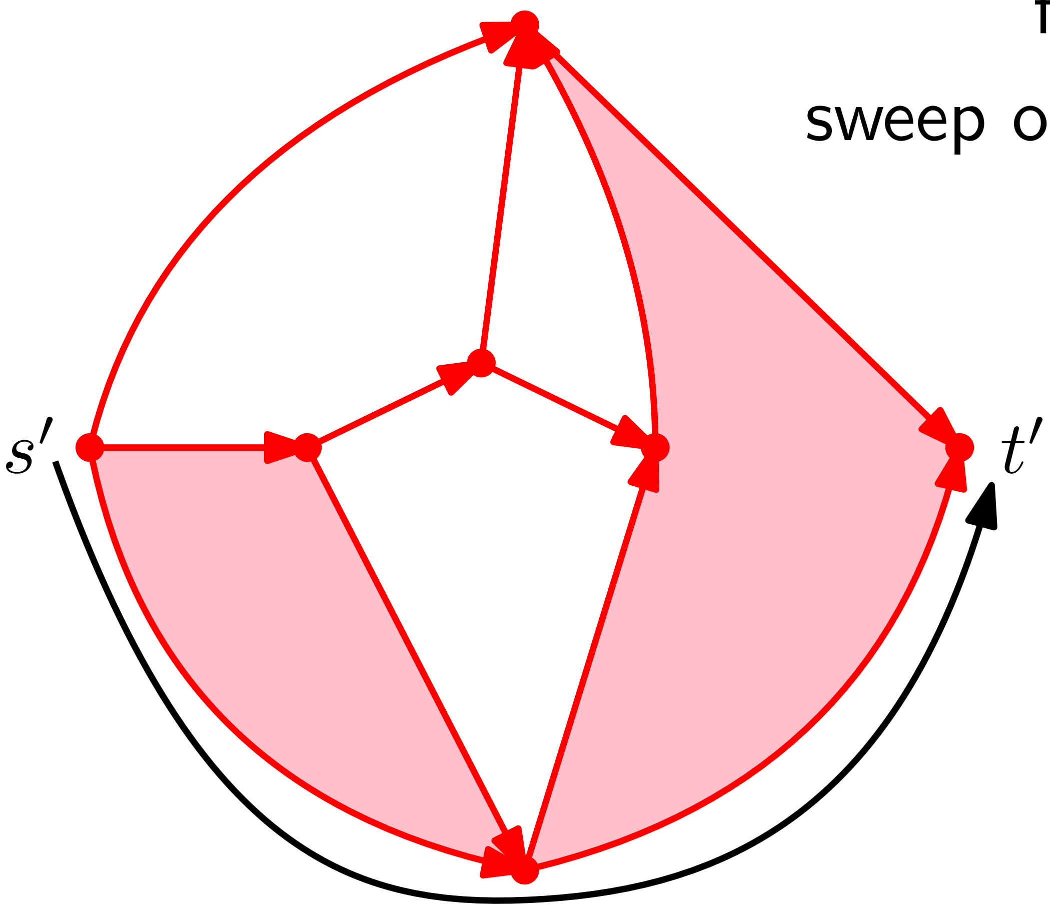
Coordinated primal-dual sweep



- sweep the dual graph with an s'–t' rope from bottom to top

sweep over the *leftmost* possible face

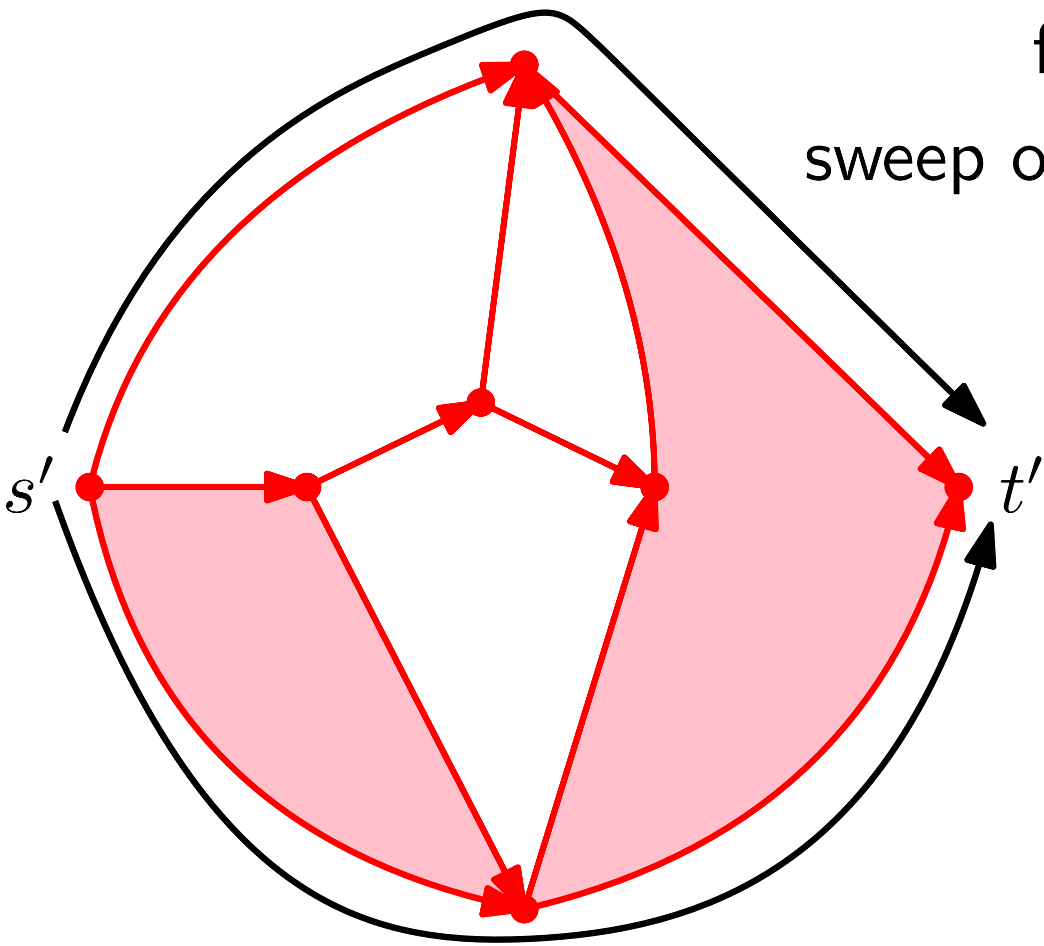
Coordinated primal-dual sweep



- sweep the dual graph with an s'–t' rope from bottom to top

sweep over the *leftmost* possible face

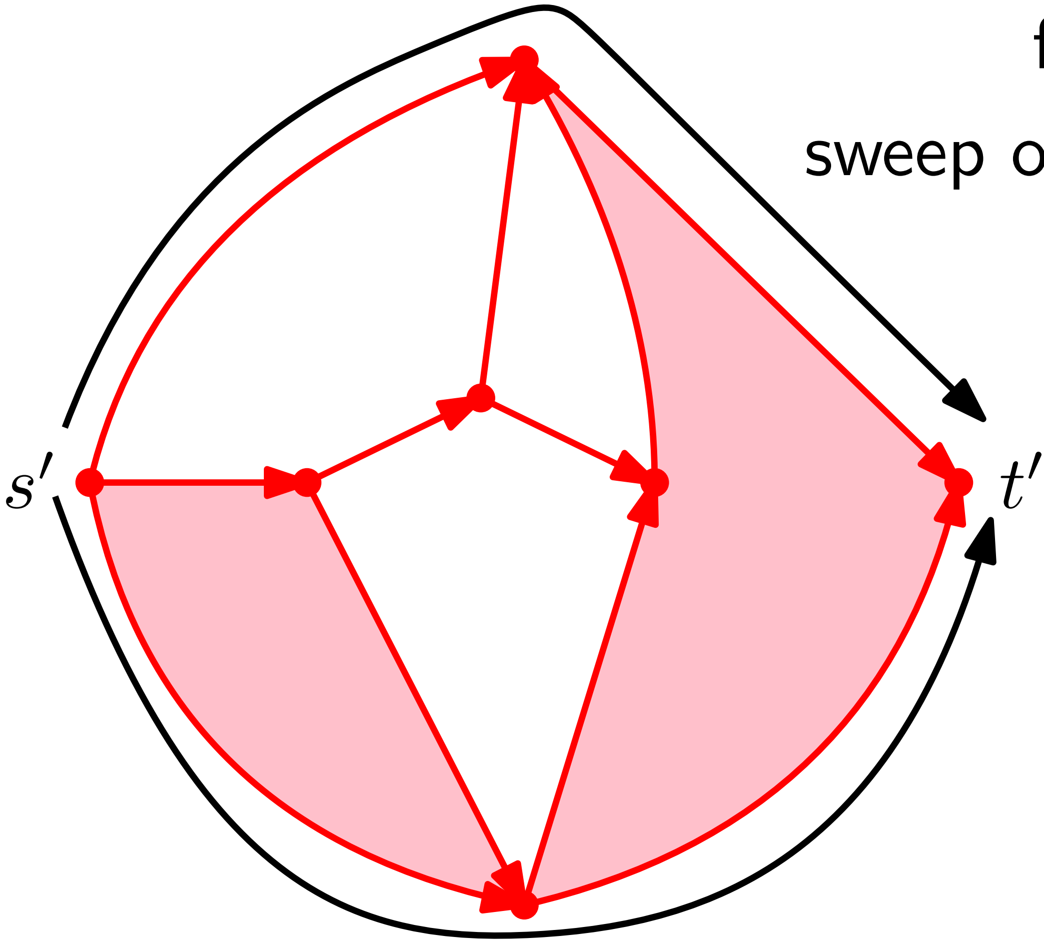
Coordinated primal-dual sweep



- sweep the dual graph with an s'–t' rope from bottom to top

sweep over the *leftmost* possible face

Coordinated primal-dual sweep

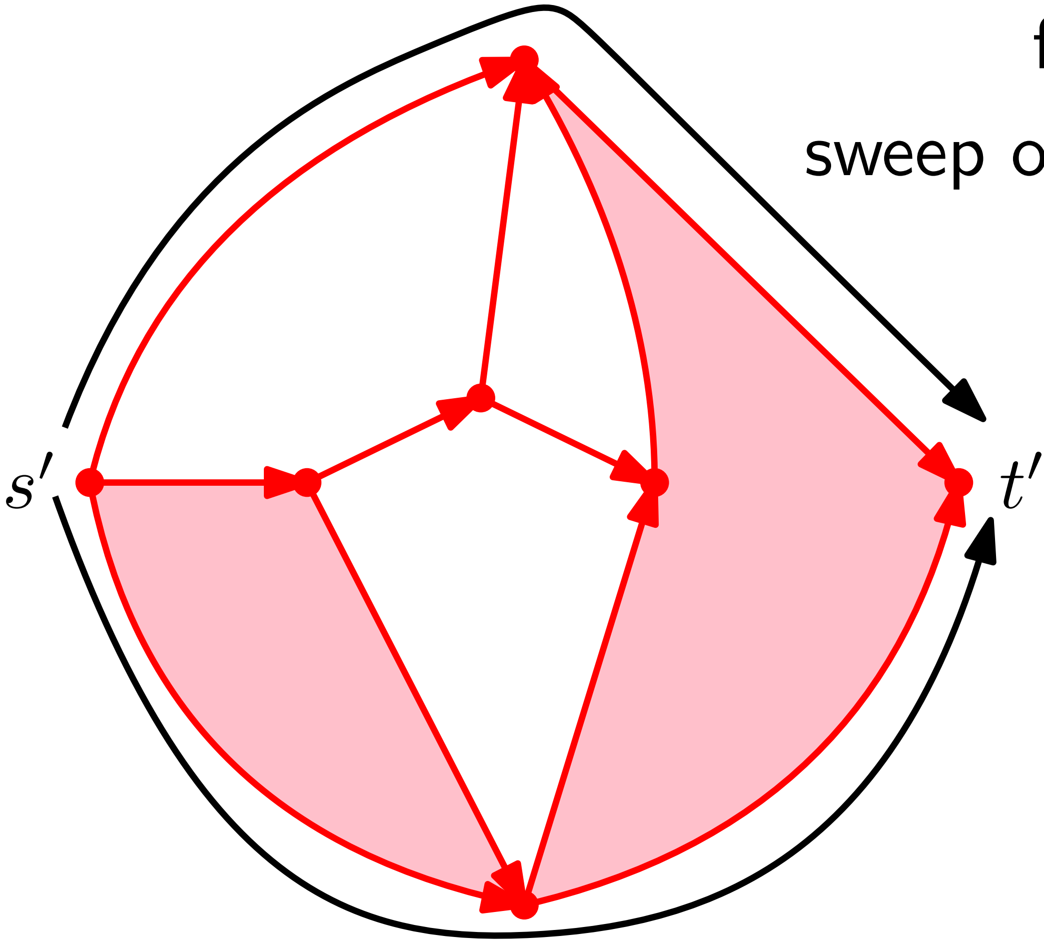


- sweep the dual graph with an s'–t' rope from bottom to top

sweep over the *leftmost* possible face

Sweep is always possible!

Coordinated primal-dual sweep

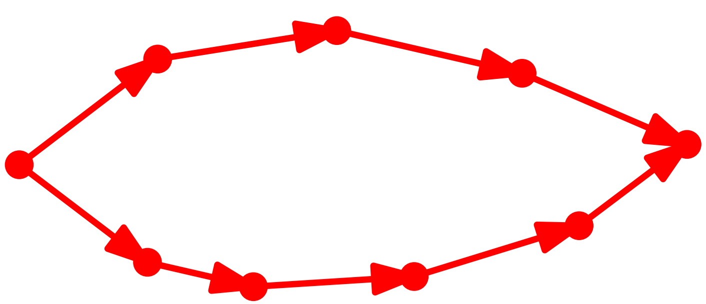


- sweep the dual graph with an s'–t' rope from bottom to top

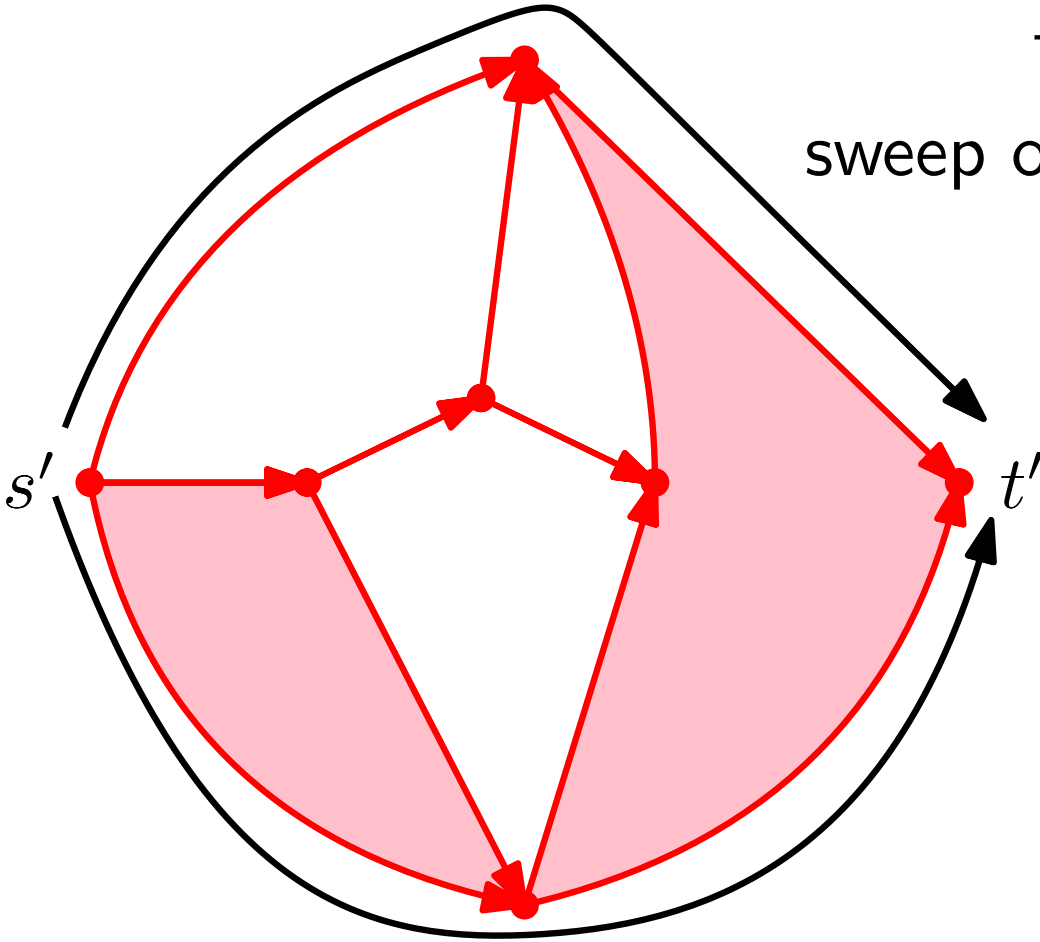
sweep over the *leftmost* possible face

Sweep is always possible!

General form of a face



Coordinated primal-dual sweep

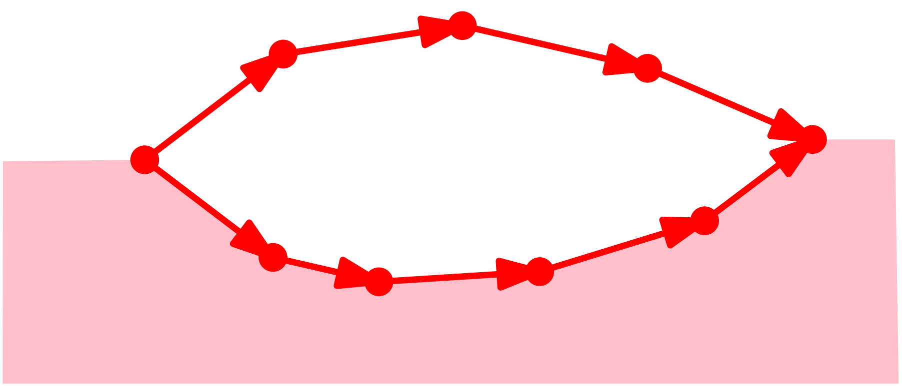


- sweep the dual graph with an s'–t' rope from bottom to top

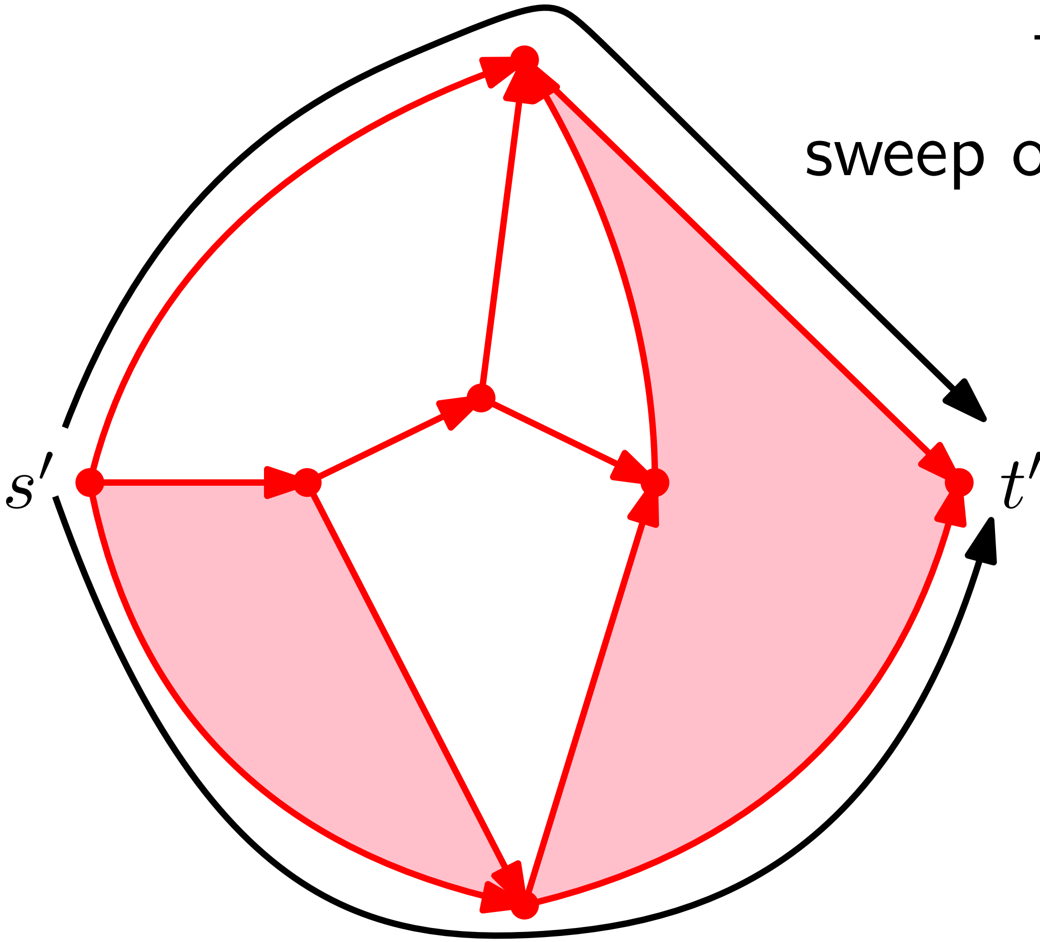
sweep over the *leftmost* possible face

Sweep is always possible!

General form of a face



Coordinated primal-dual sweep

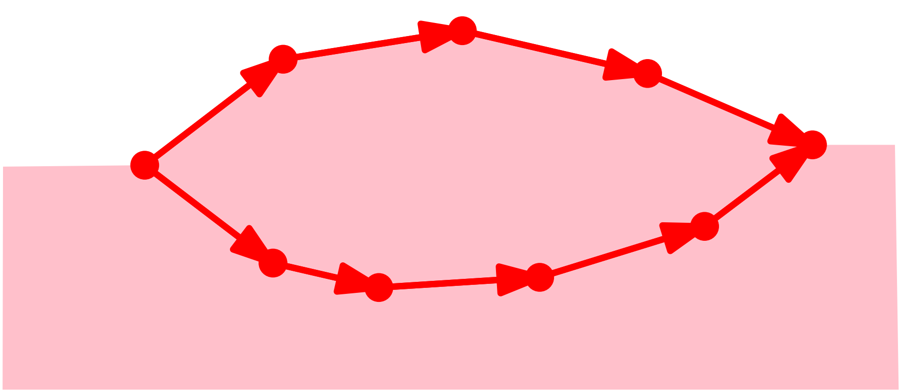


- sweep the dual graph with an s'–t' rope from bottom to top

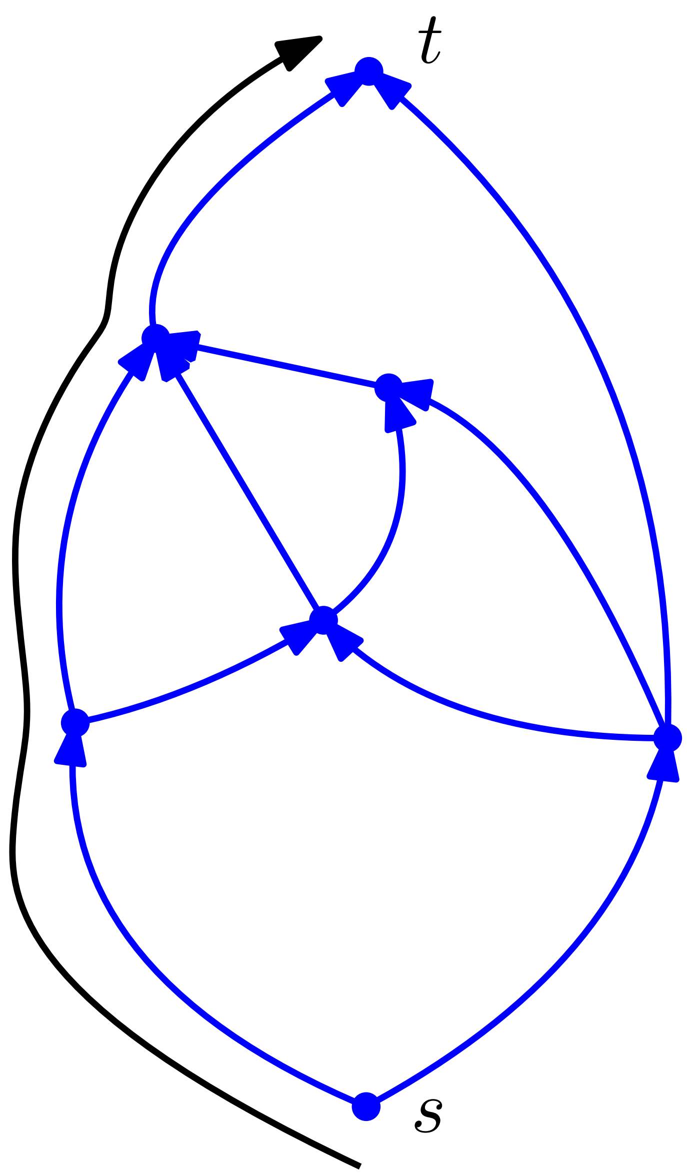
sweep over the *leftmost* possible face

Sweep is always possible!

General form of a face

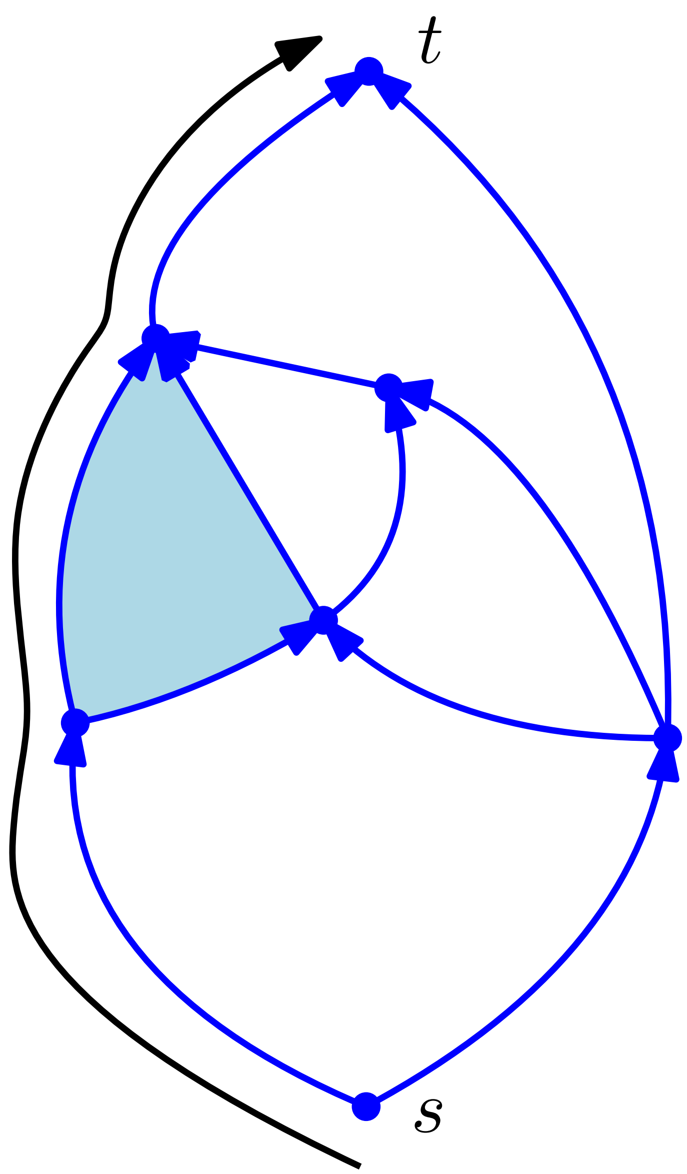


Coordinated primal-dual sweep



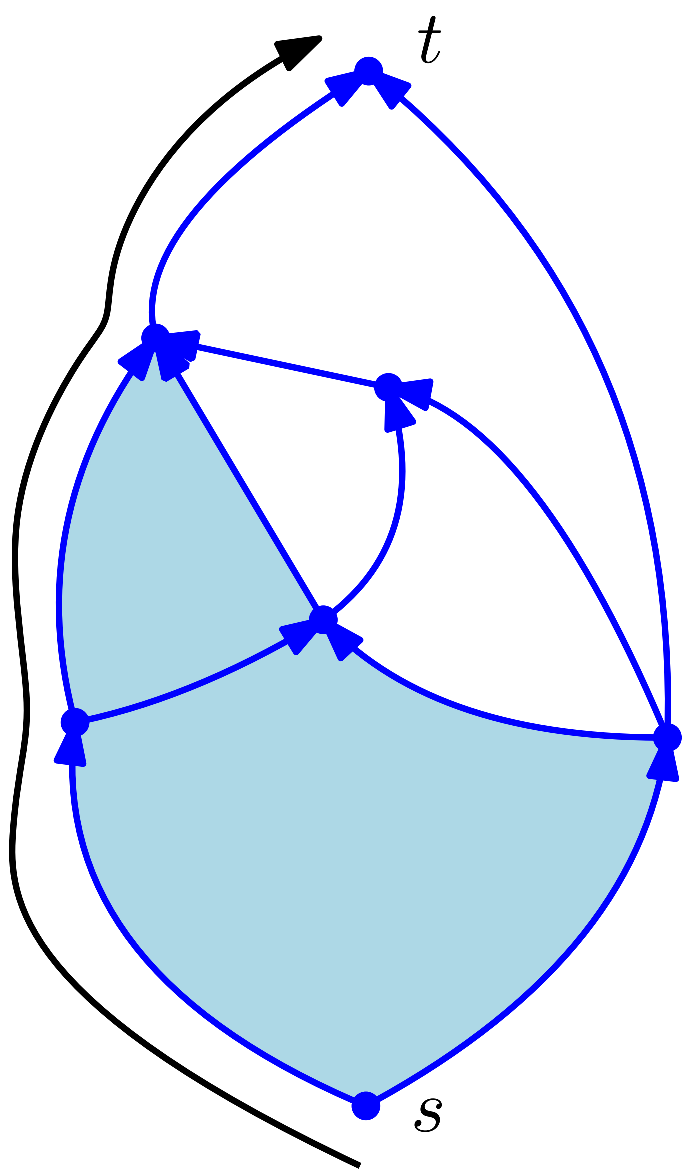
- sweep the primal graph with an s–t rope from left to right

Coordinated primal-dual sweep



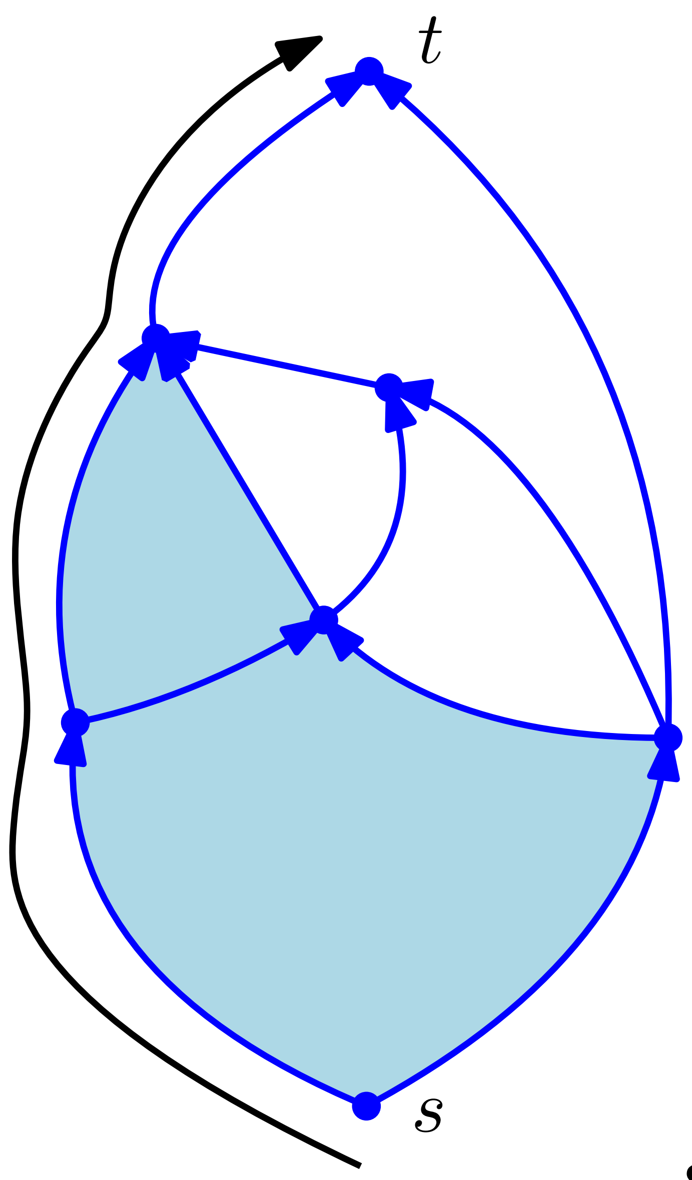
- sweep the primal graph with an s–t rope from left to right

Coordinated primal-dual sweep



- sweep the primal graph with an s–t rope from left to right

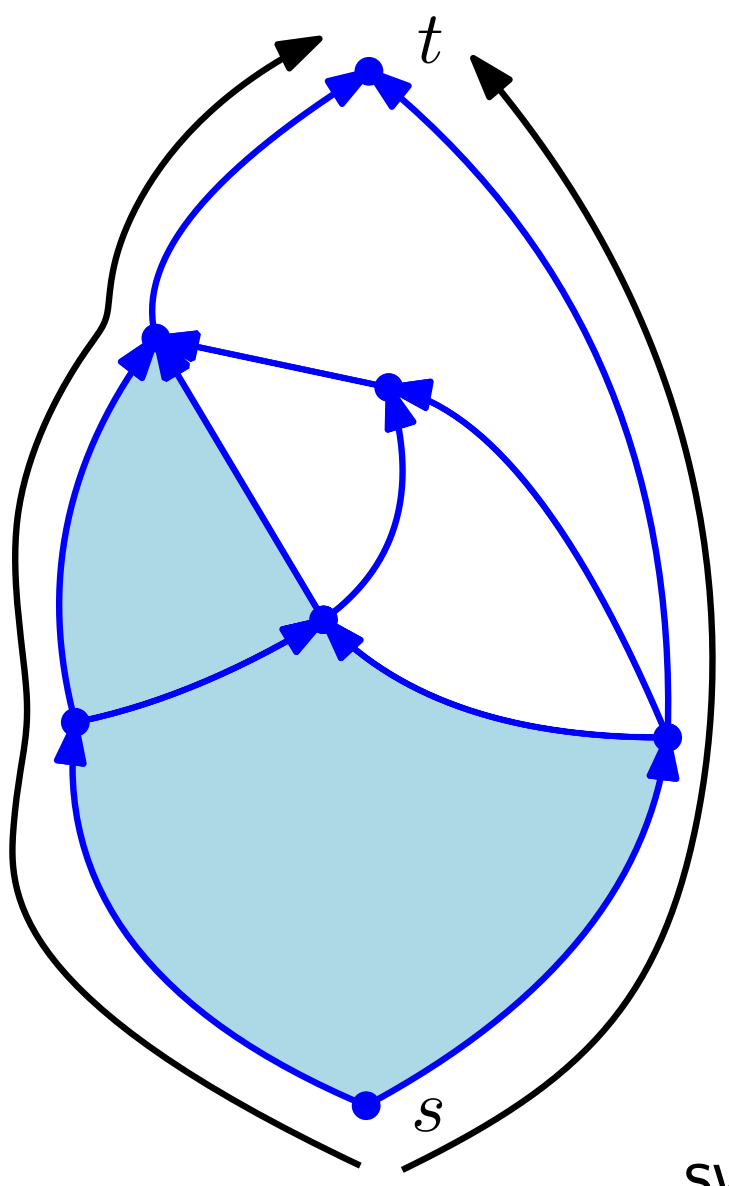
Coordinated primal-dual sweep



- sweep the primal graph with an s–t rope from left to right

sweep over the *lowest* possible face

Coordinated primal-dual sweep

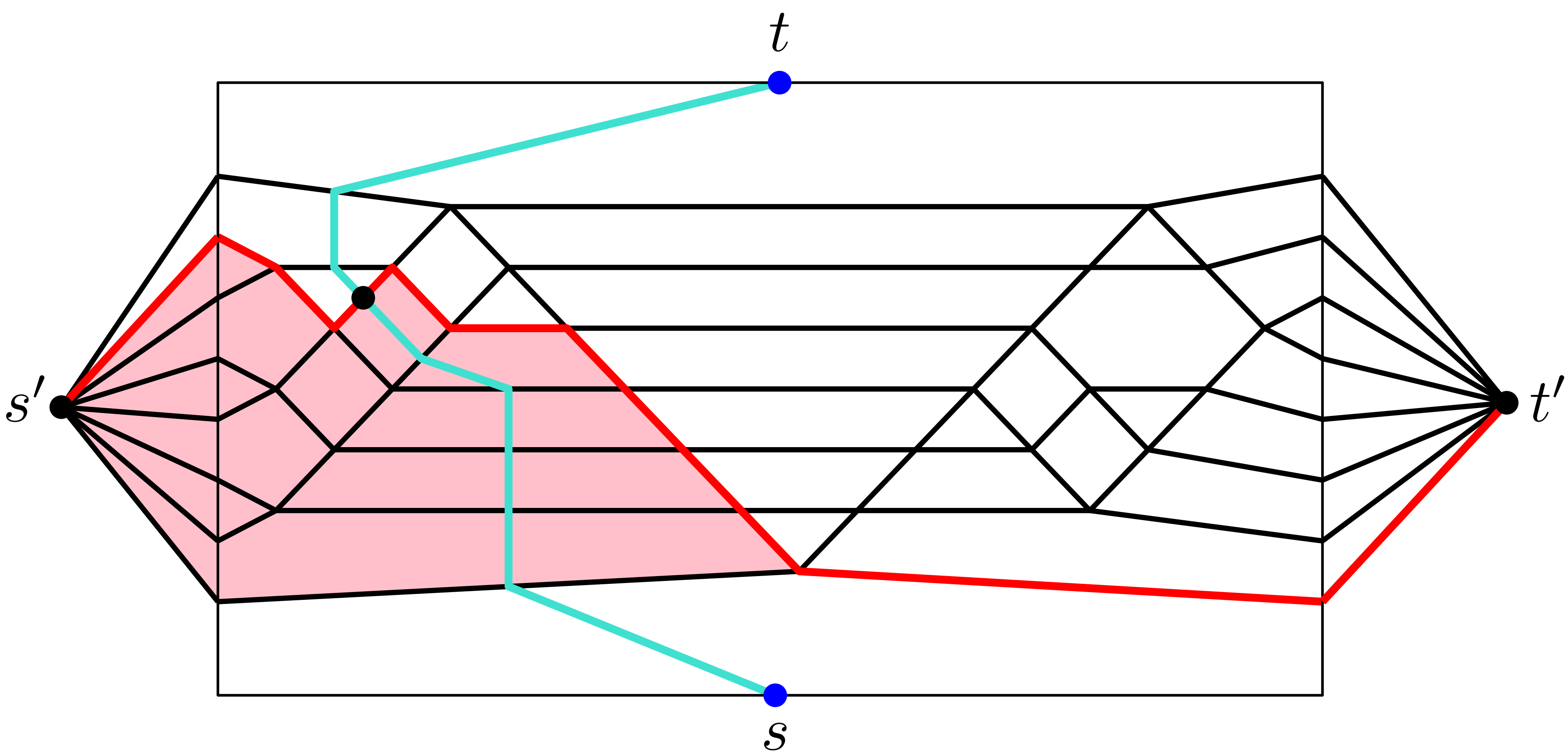


- sweep the primal graph with an s–t rope from left to right

sweep over the *lowest* possible face

Animation

Freie Universität Berlin

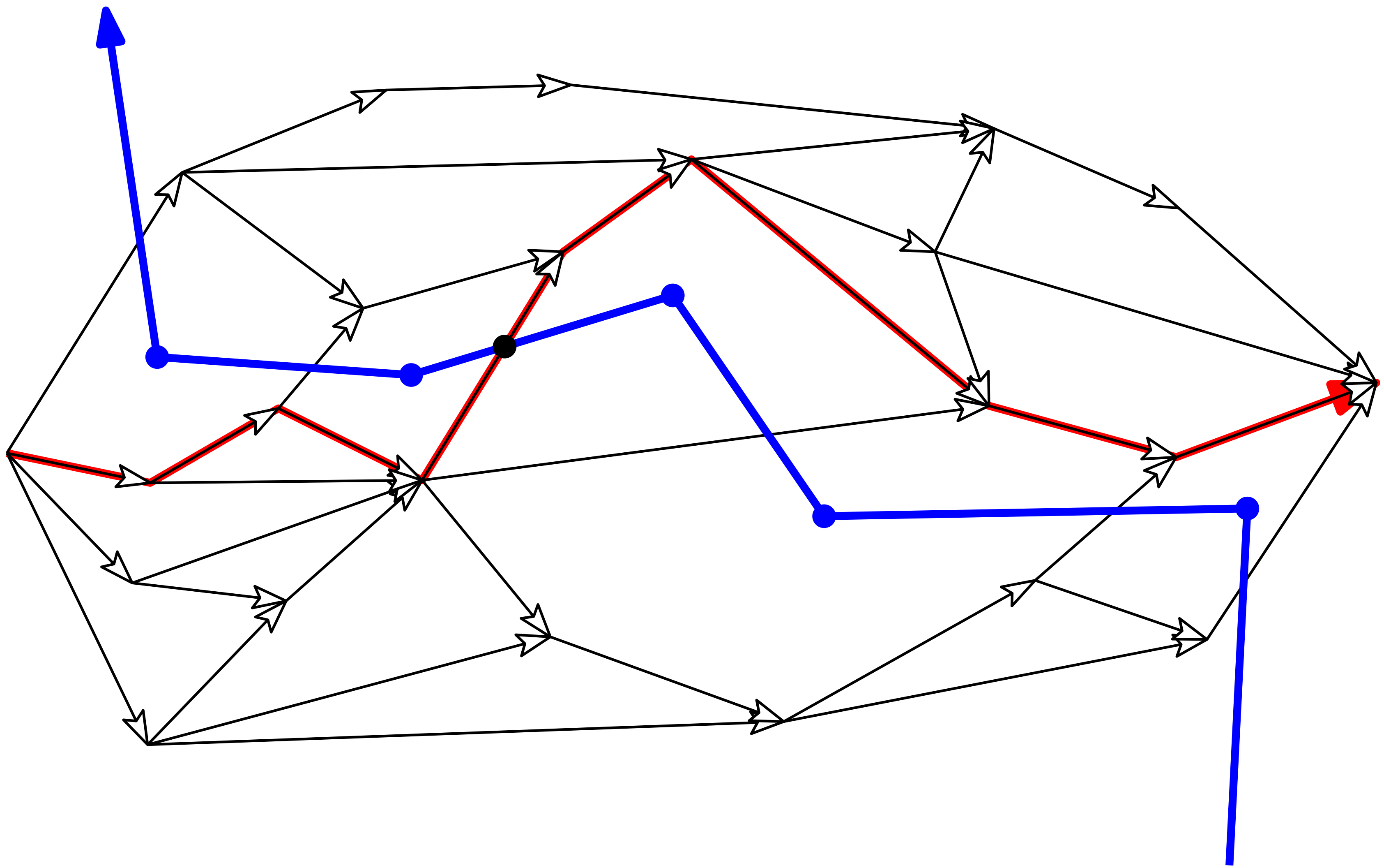


dual rope in the dual (multi-)graph

primal rope (The primal graph is not shown.)

page.mi.fu-berlin.de/rote/Papers/slides/Wuerzburg-2020-Simultaneous-sweep-Animation.pdf

A snapshot



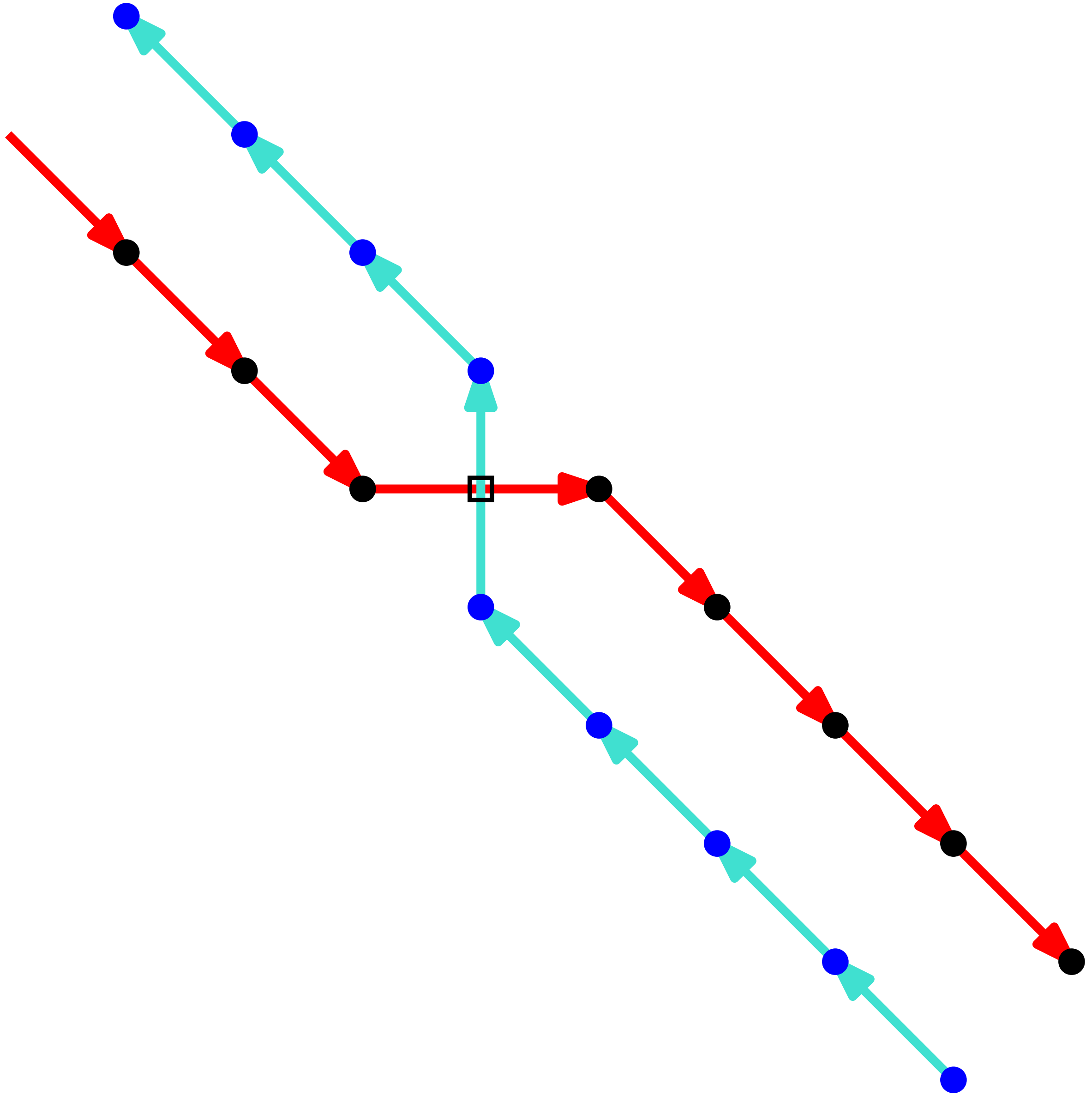
Coordinated sweep

There is a (unique) coordinated primal-dual sweep with the following properties:

- The primal rope always crosses the dual rope exactly once.
- The primal and the dual rope stay "close" to each other.
- Exactly one rope can advance, depending on the situation at the crossing.
- Every primal-dual edge pair is visited exactly once.
- Each individual sweep is a leftmost/bottommost sweep.

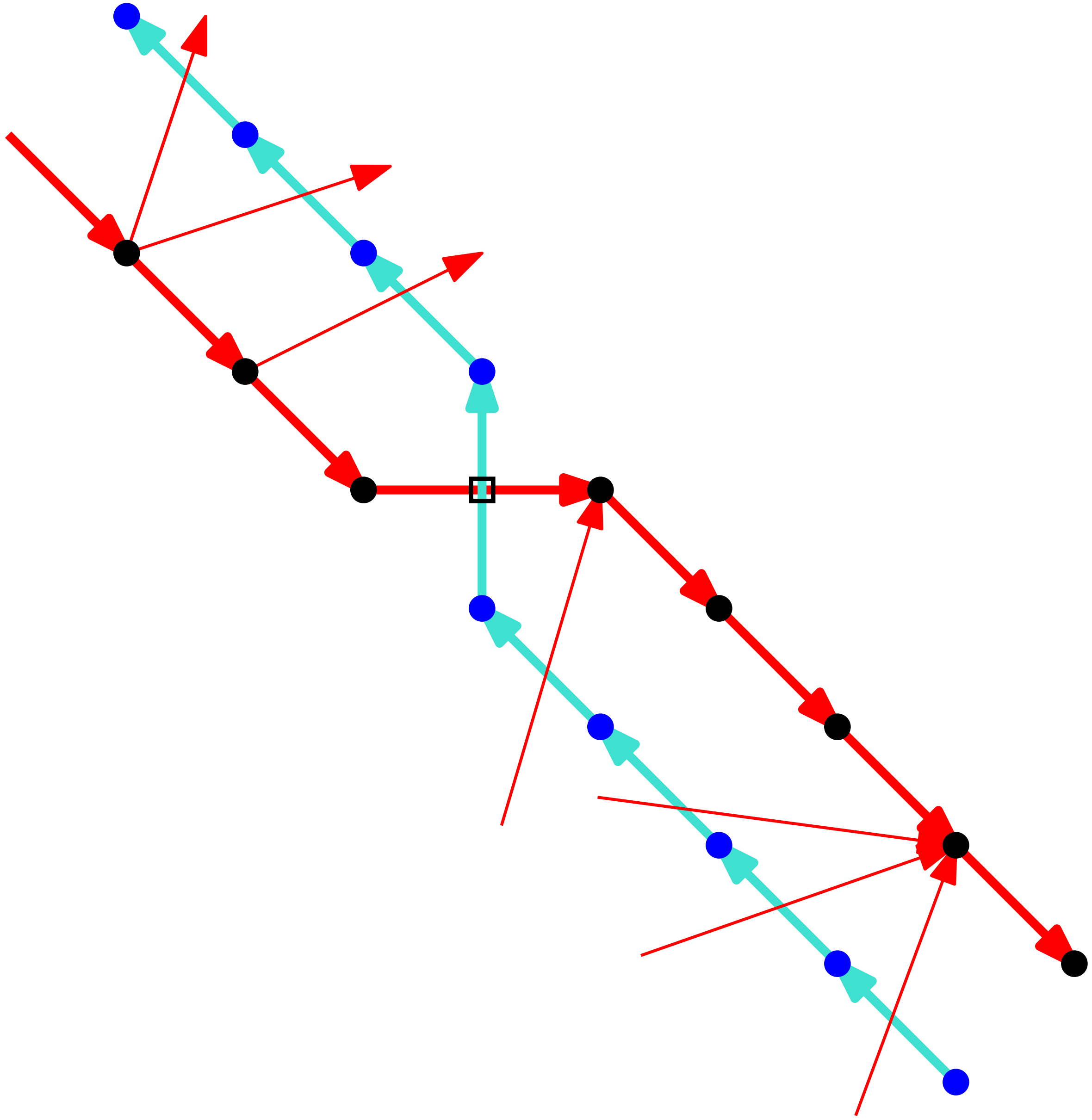
Coordinated sweep

general situation:



Coordinated sweep

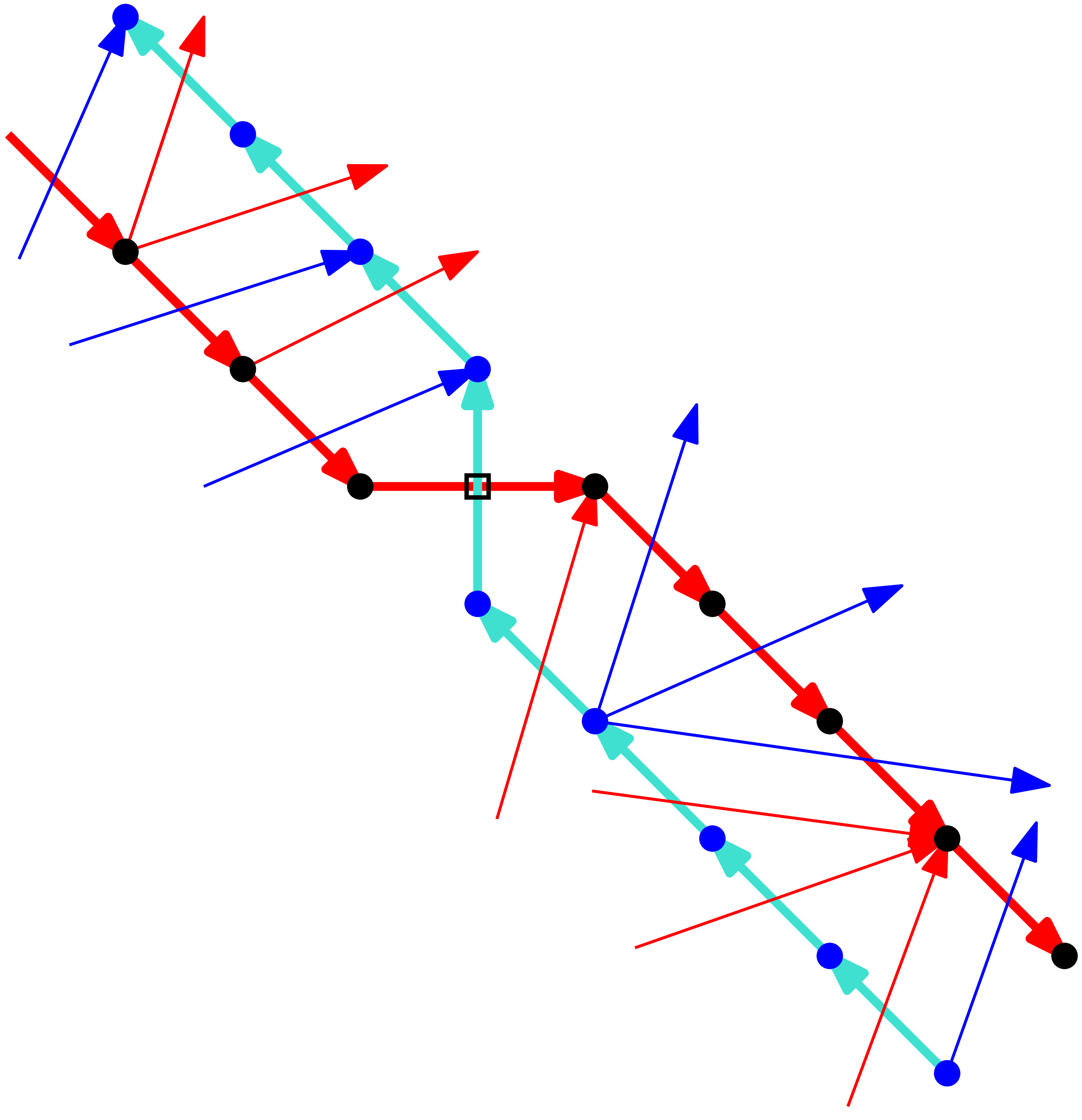
general situation:



Coordinated sweep

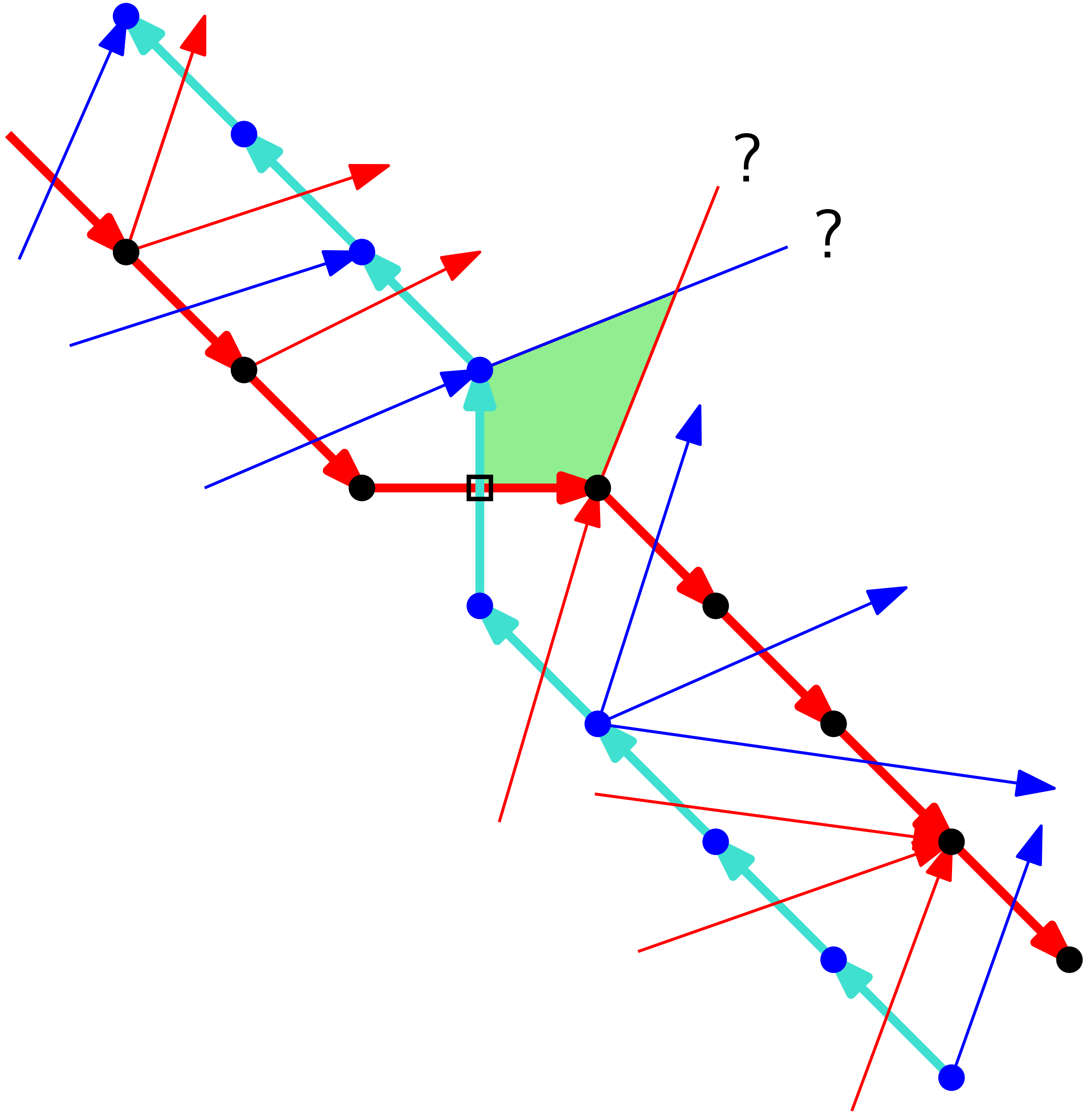
Freie Universität Berlin

general situation:



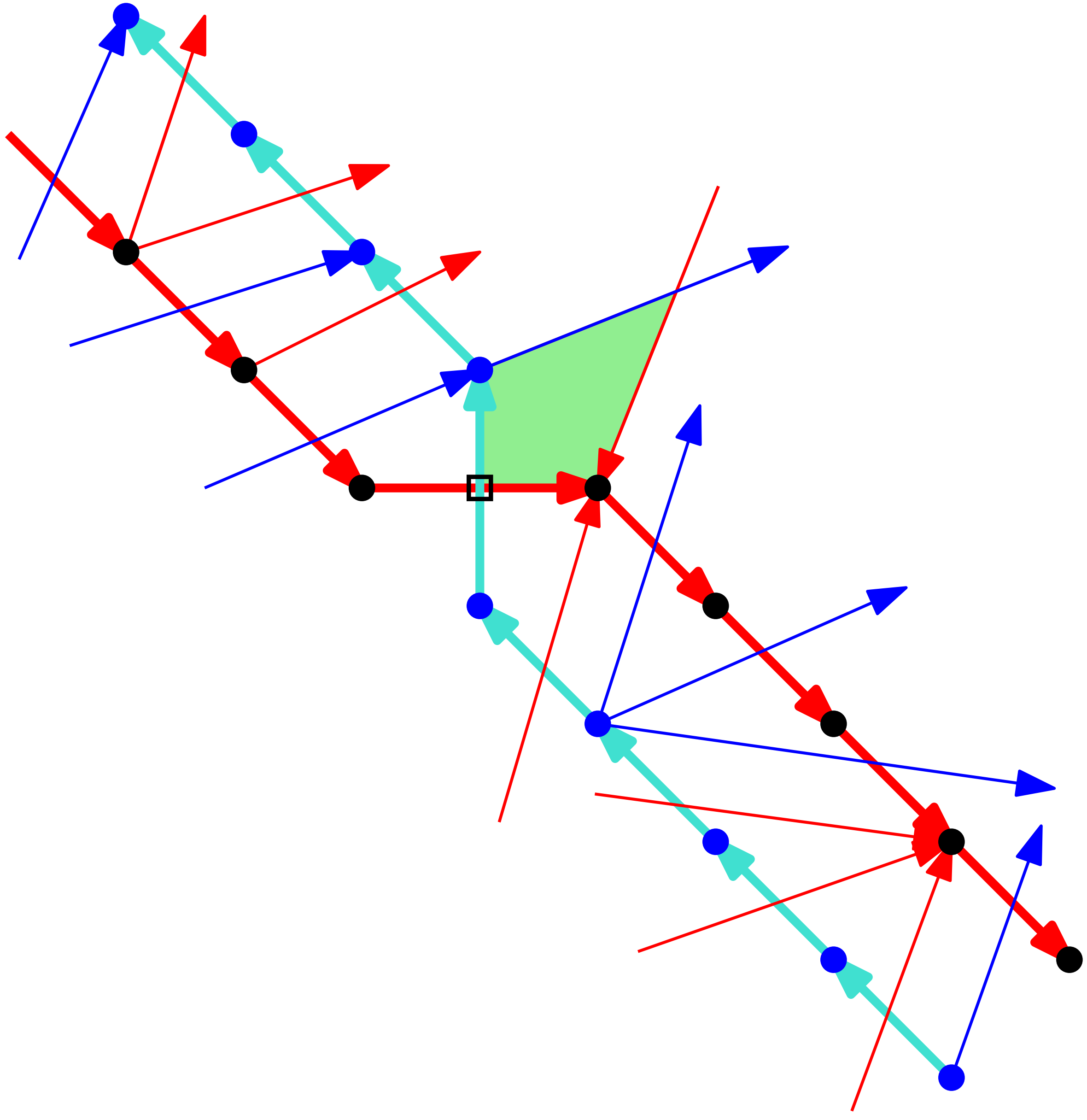
Coordinated sweep

general situation:



Coordinated sweep

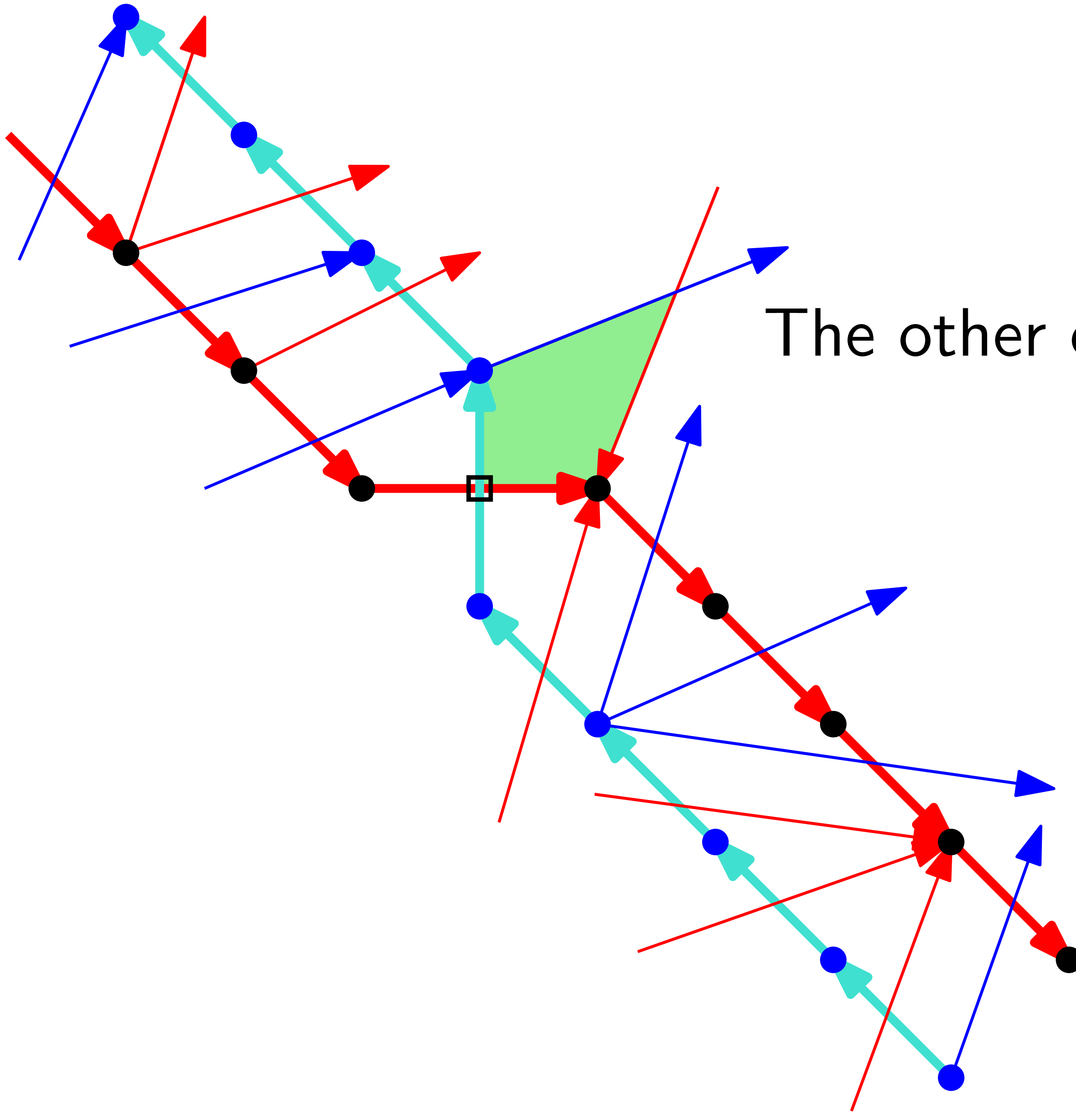
general situation:



Coordinated sweep

Freie Universität Berlin

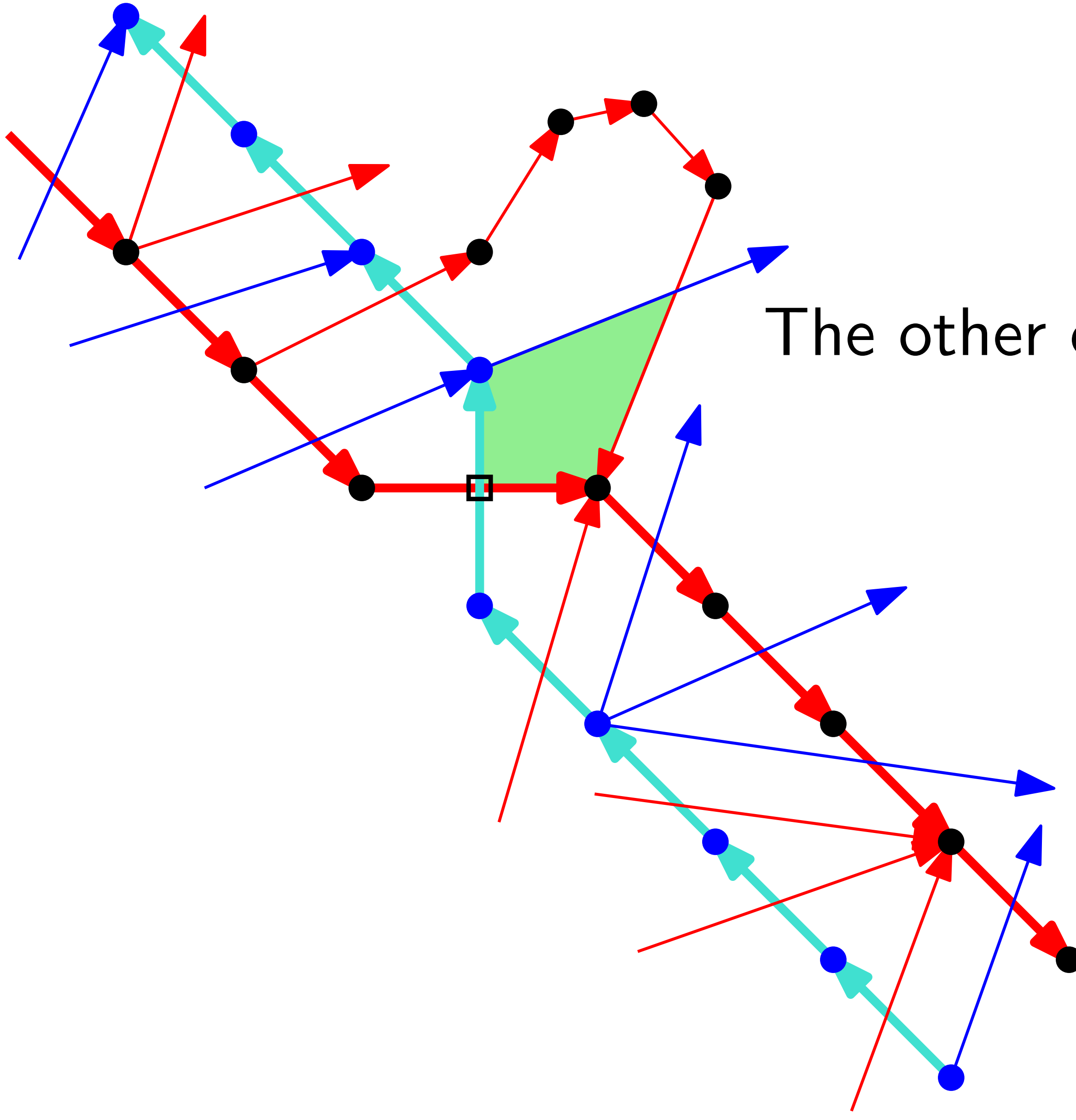
general situation:



The other case is symmetric.

Coordinated sweep

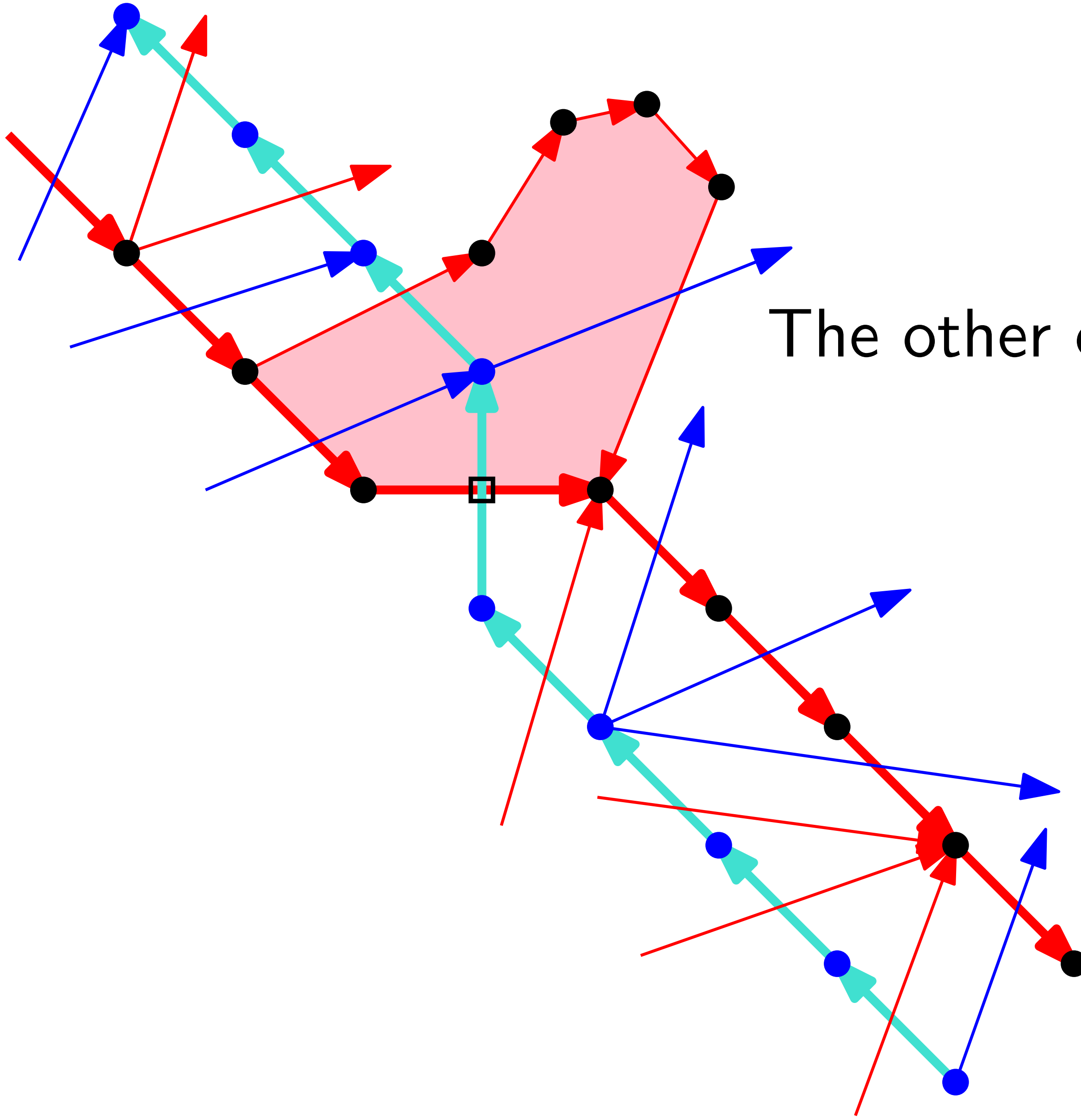
general situation:



The other case is symmetric.

Coordinated sweep

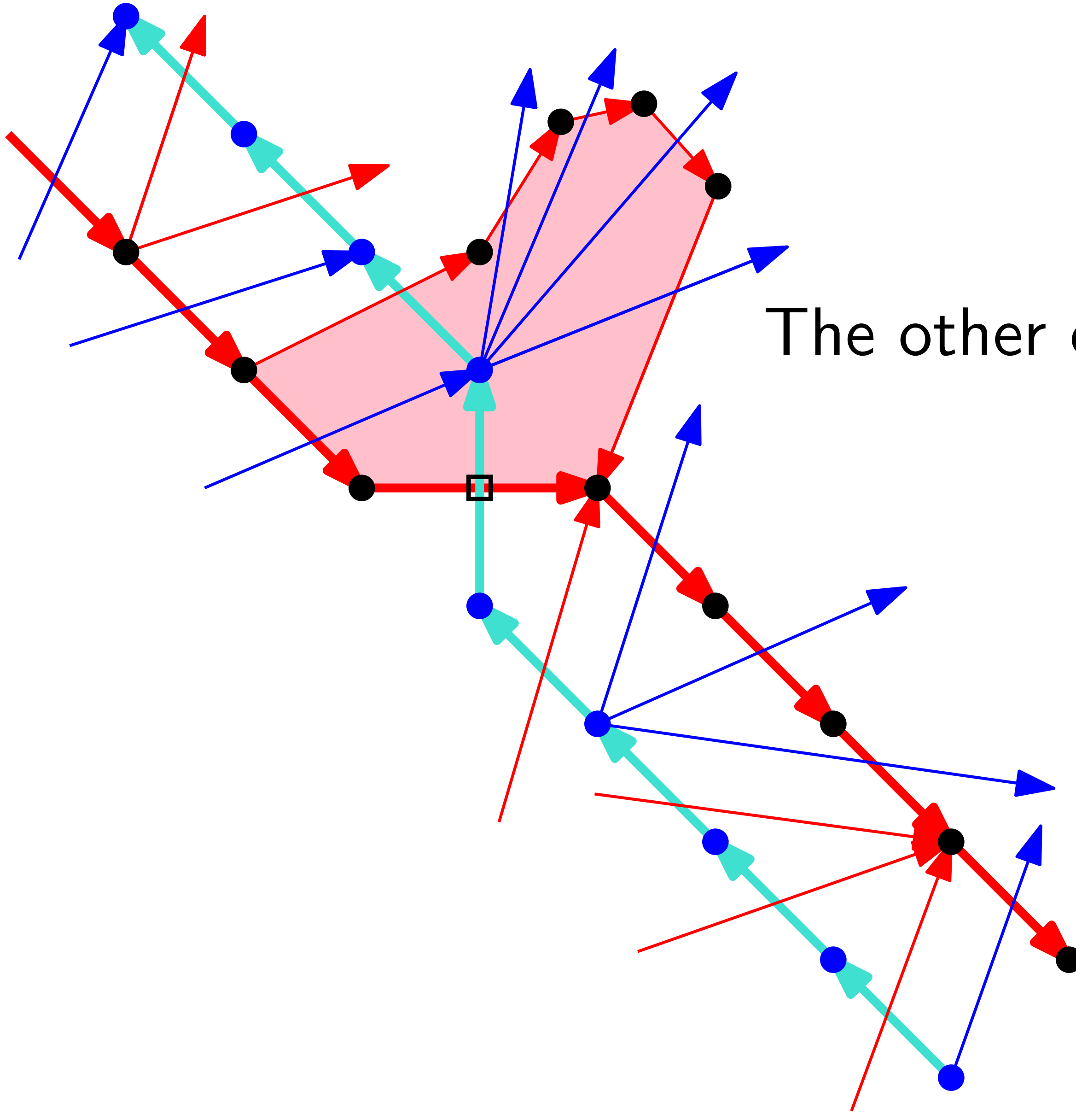
general situation:



The other case is symmetric.

Coordinated sweep

general situation:



The other case is symmetric.

Coordinated sweep

Questions:

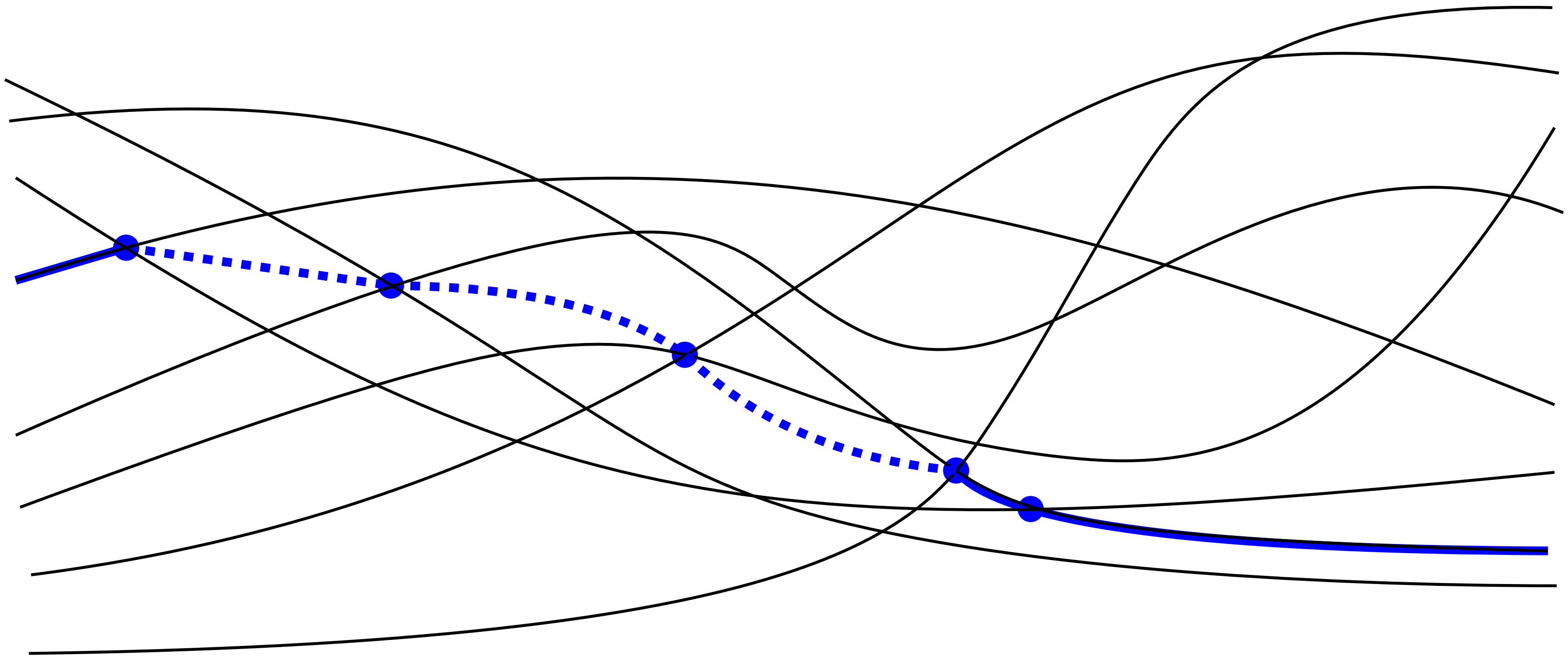
- Consider the (primal/dual) rope length:

  In terms of which parameters can it be bounded?

- Consider a primal sweep in which several independent faces can be swept simultaneously:

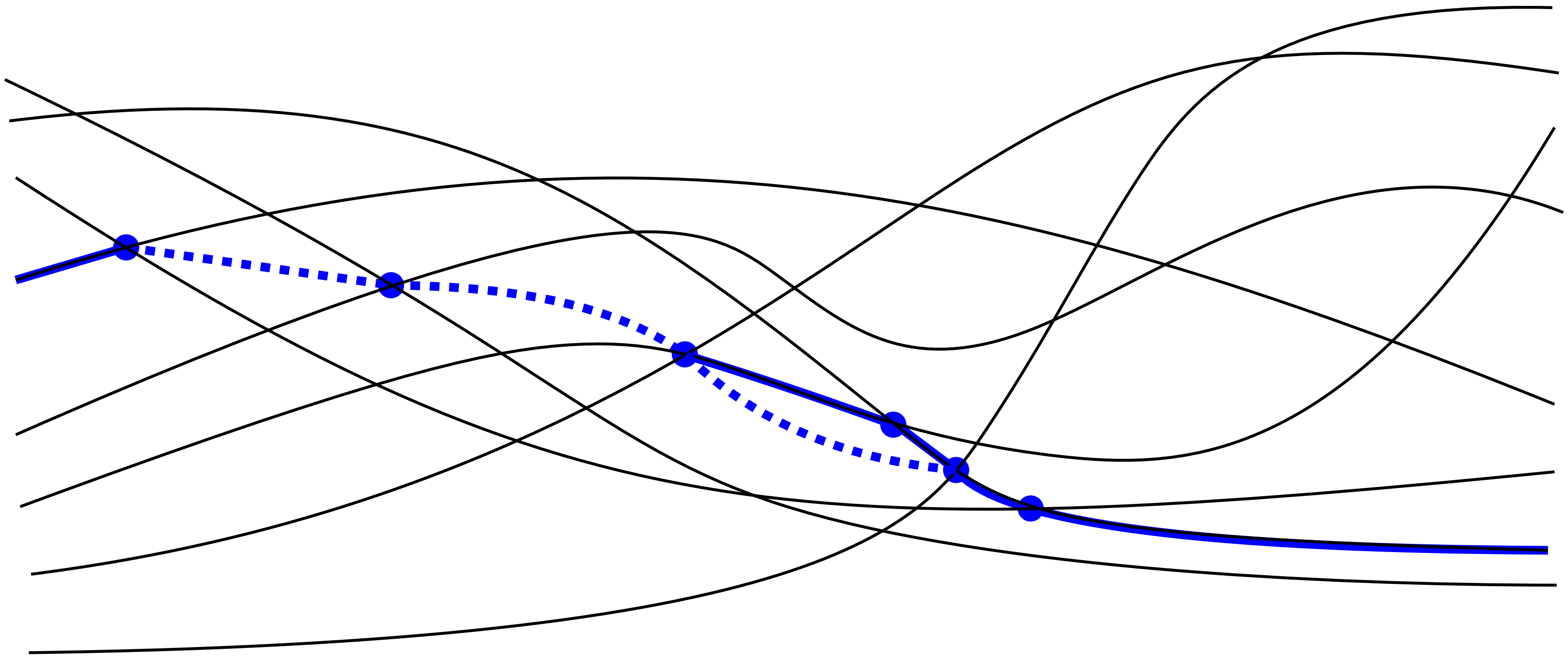
  Can this reduce the required rope length?

What really matters in practice



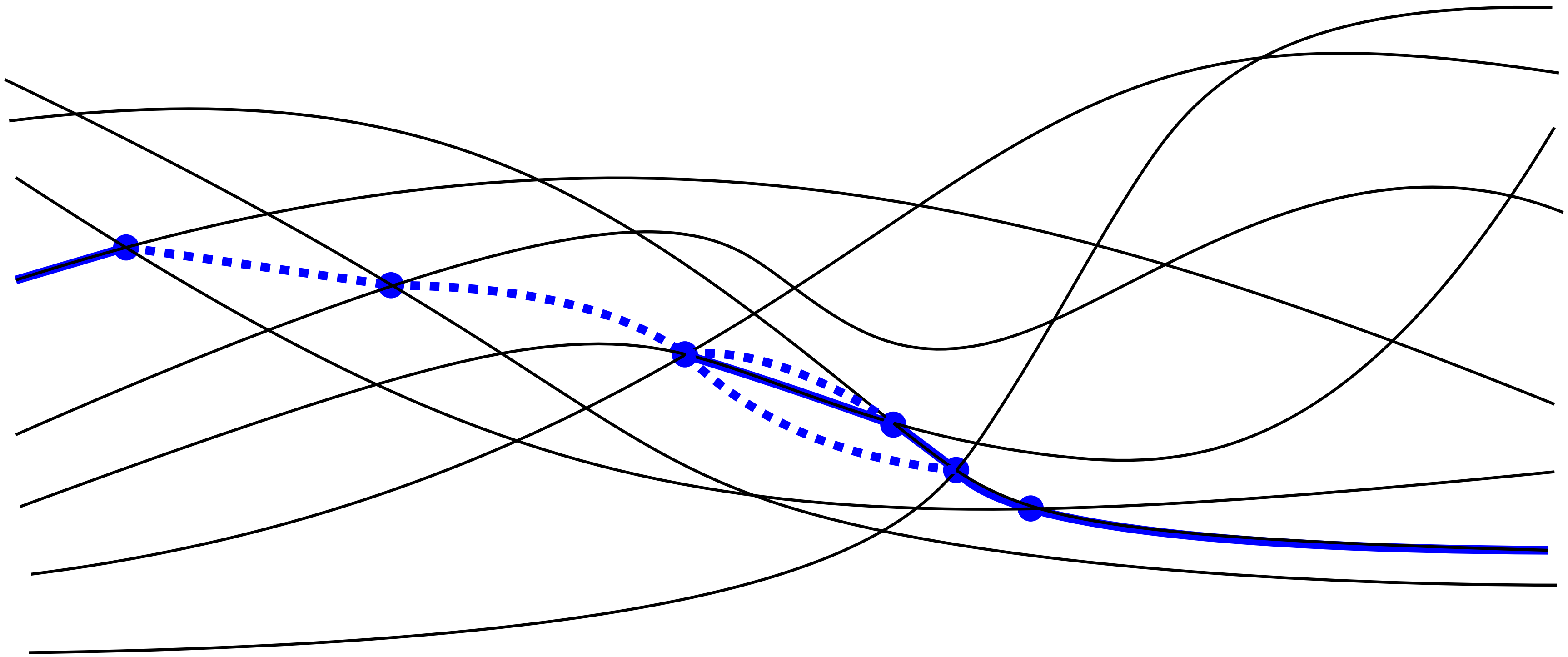
- Several *distribute* steps are done simultaneously, followed by *collect*s
- *cross* steps are done individually

What really matters in practice



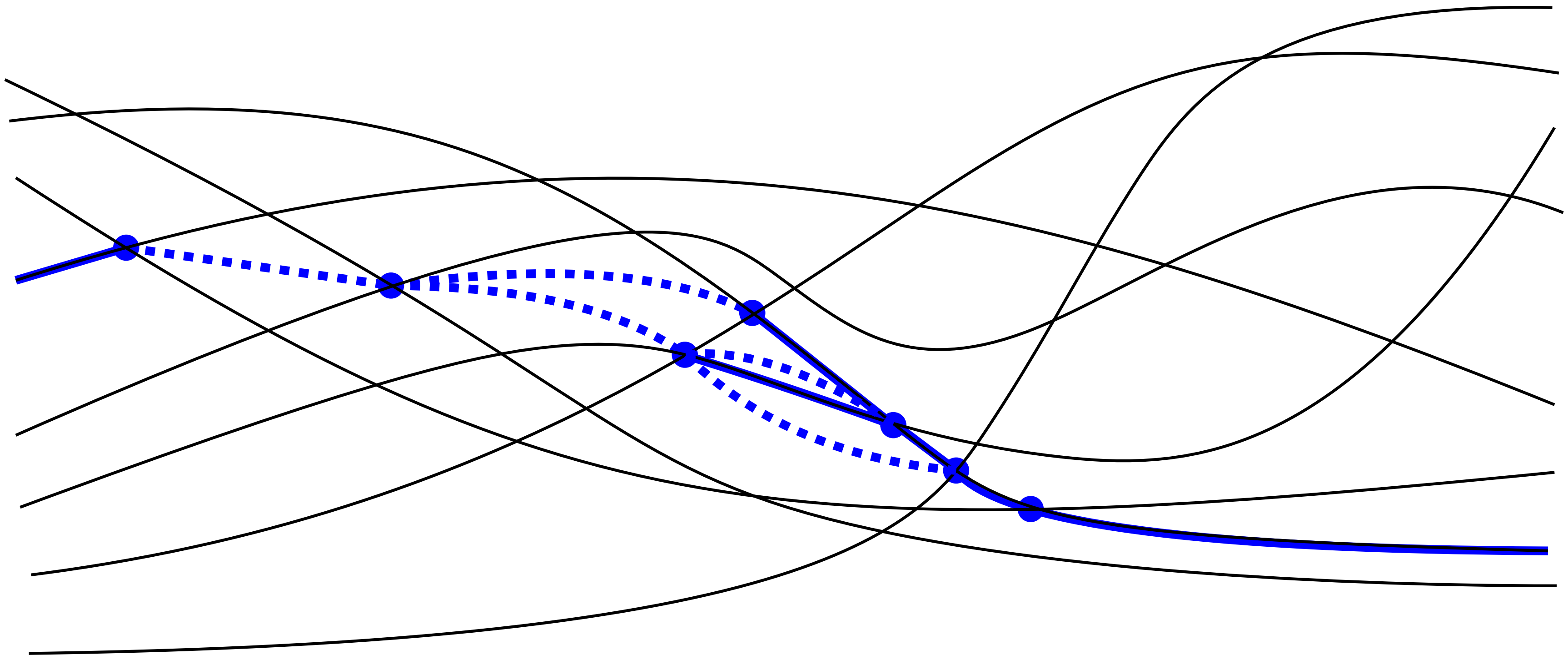
- Several *distribute* steps are done simultaneously, followed by *collects*
- *cross* steps are done individually

What really matters in practice



- Several *distribute* steps are done simultaneously, followed by *collects*
- *cross* steps are done individually

What really matters in practice



- Several *distribute* steps are done simultaneously, followed by *collects*
- *cross* steps are done individually