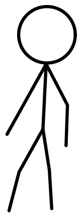
Tree Drawings with Columns

Jonathan Klawitter · Johannes Zink

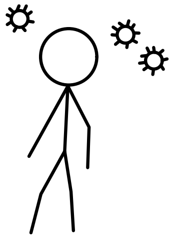
THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

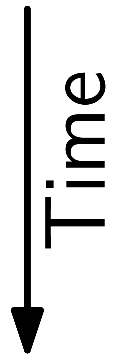
Motivation

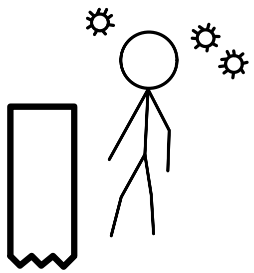


Motivation

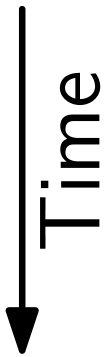


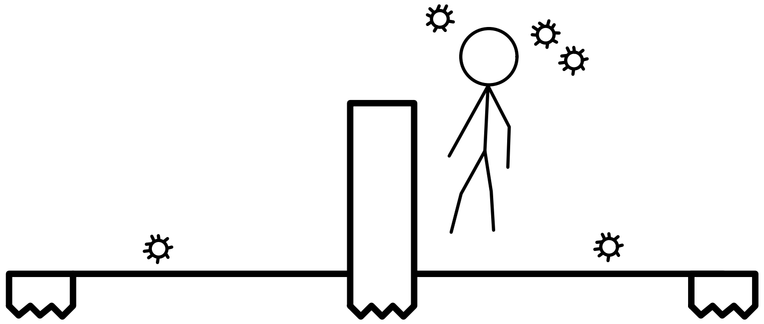
Motivation



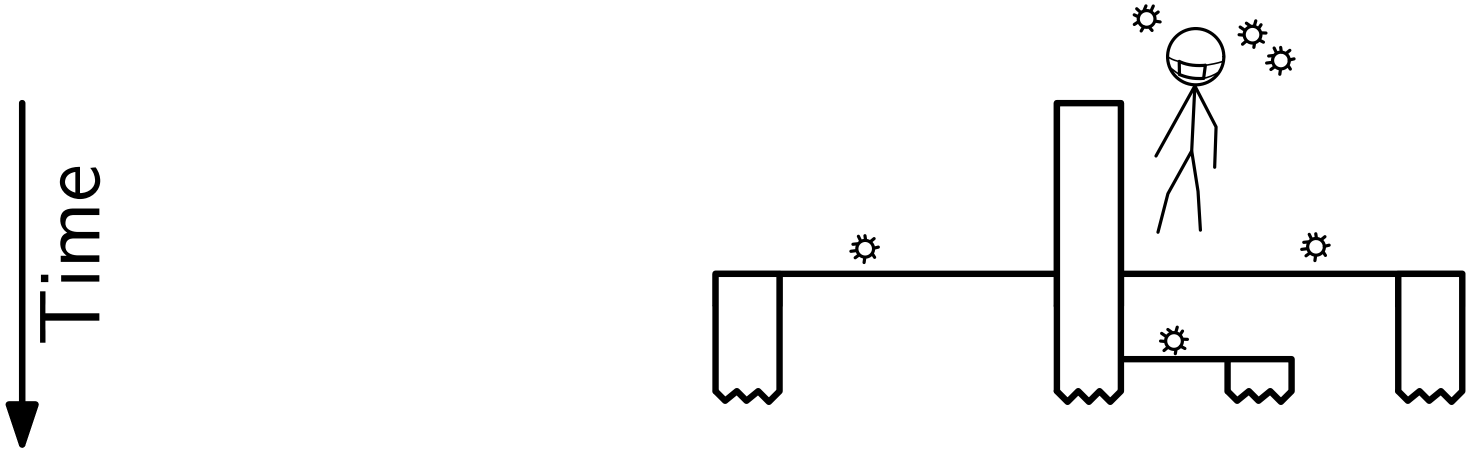


Motivation

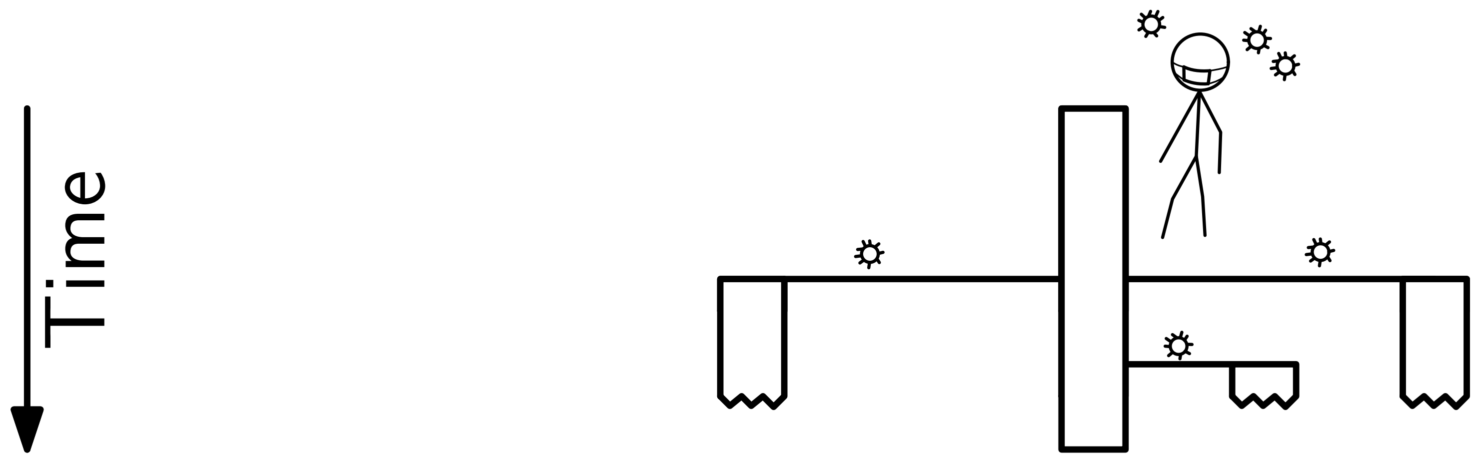




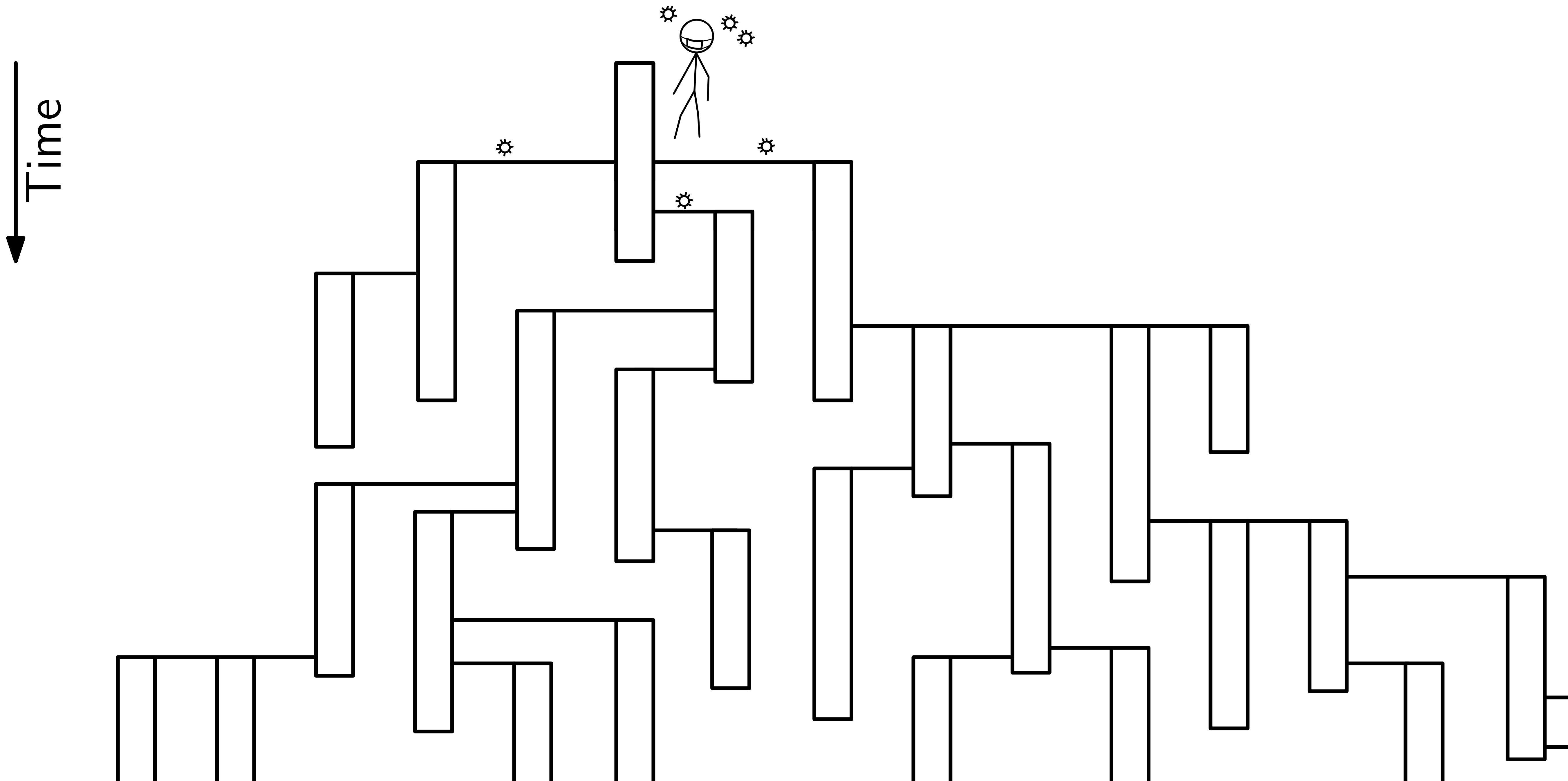
Motivation



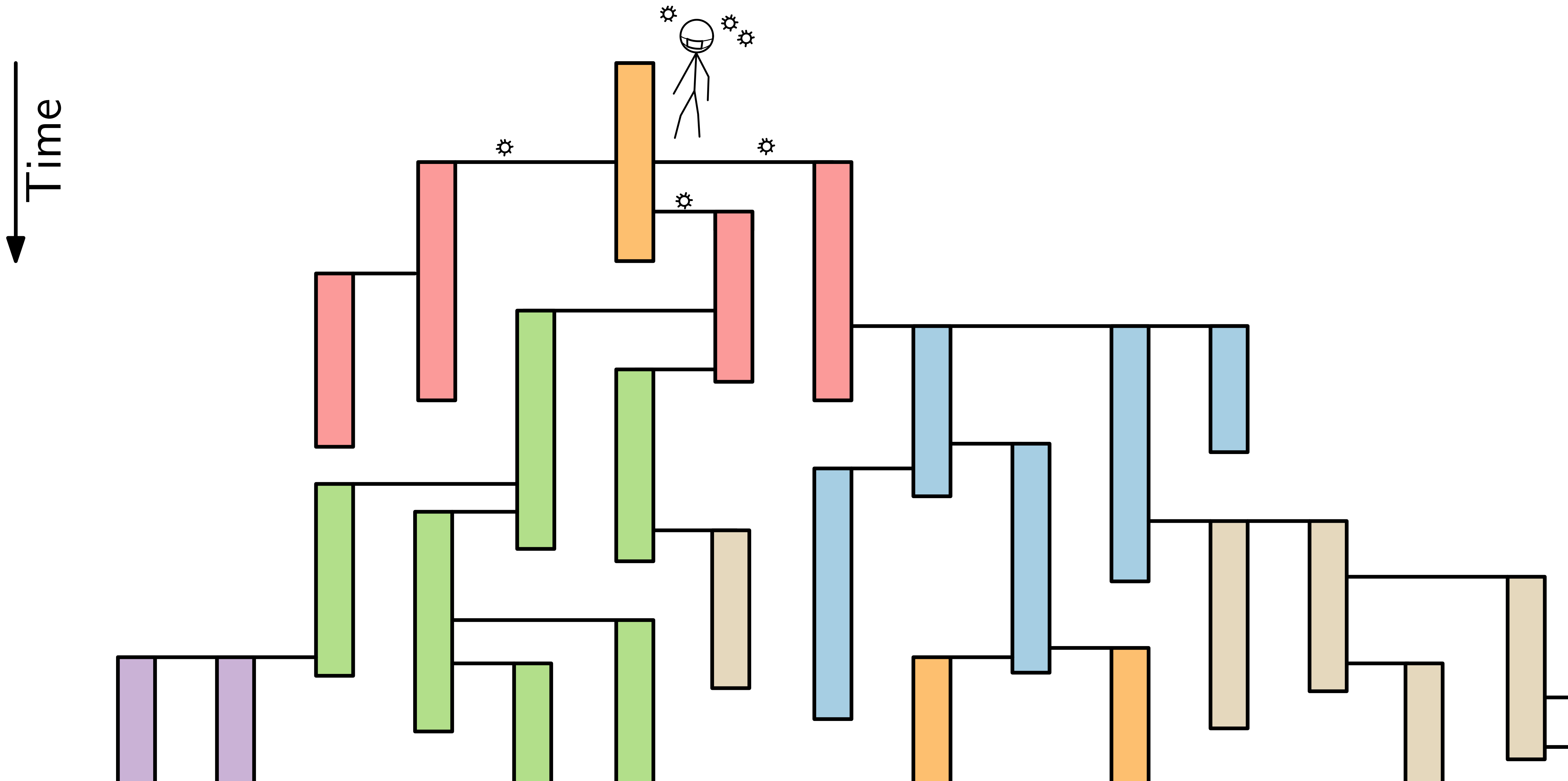
Motivation



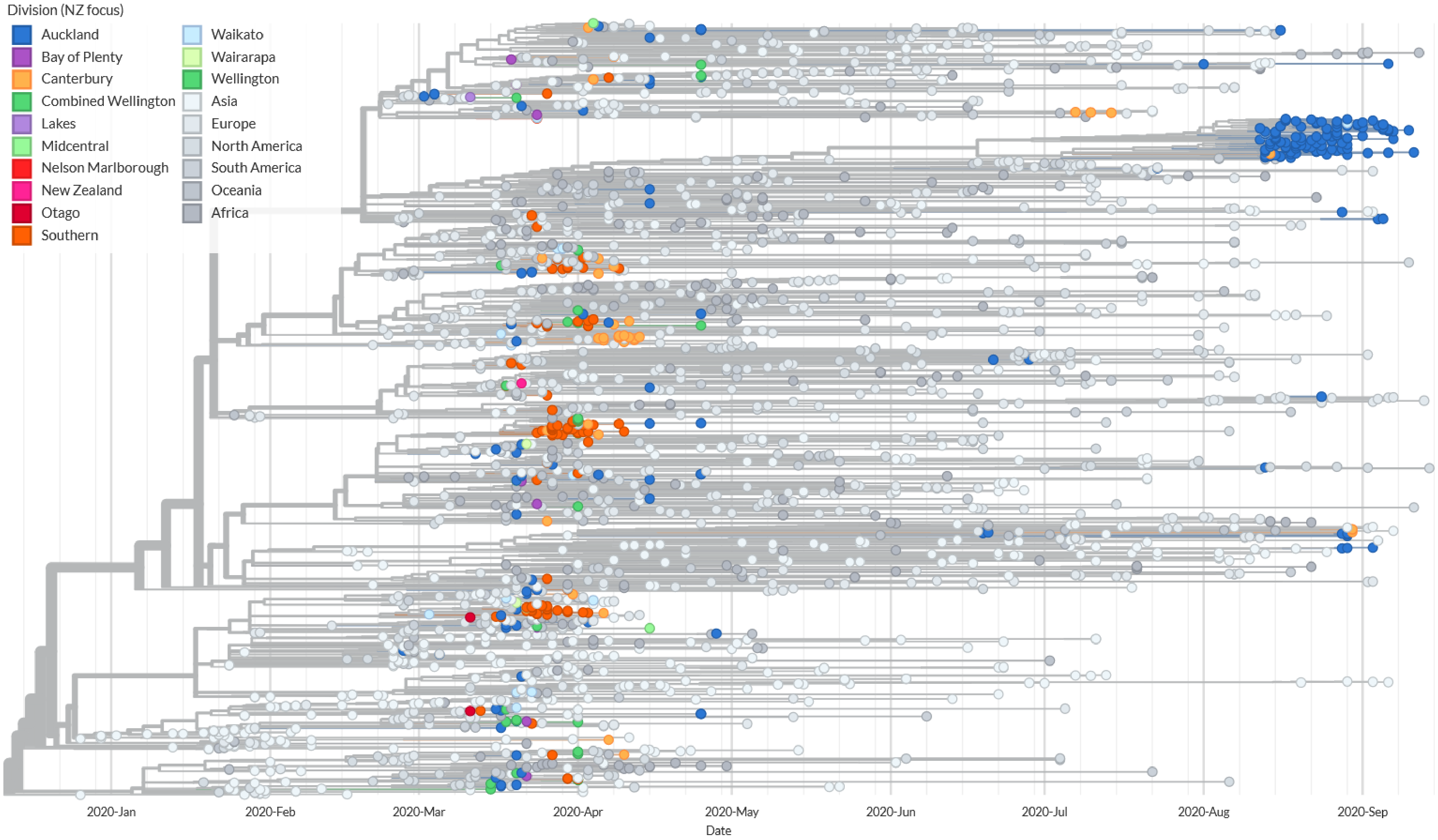
Motivation



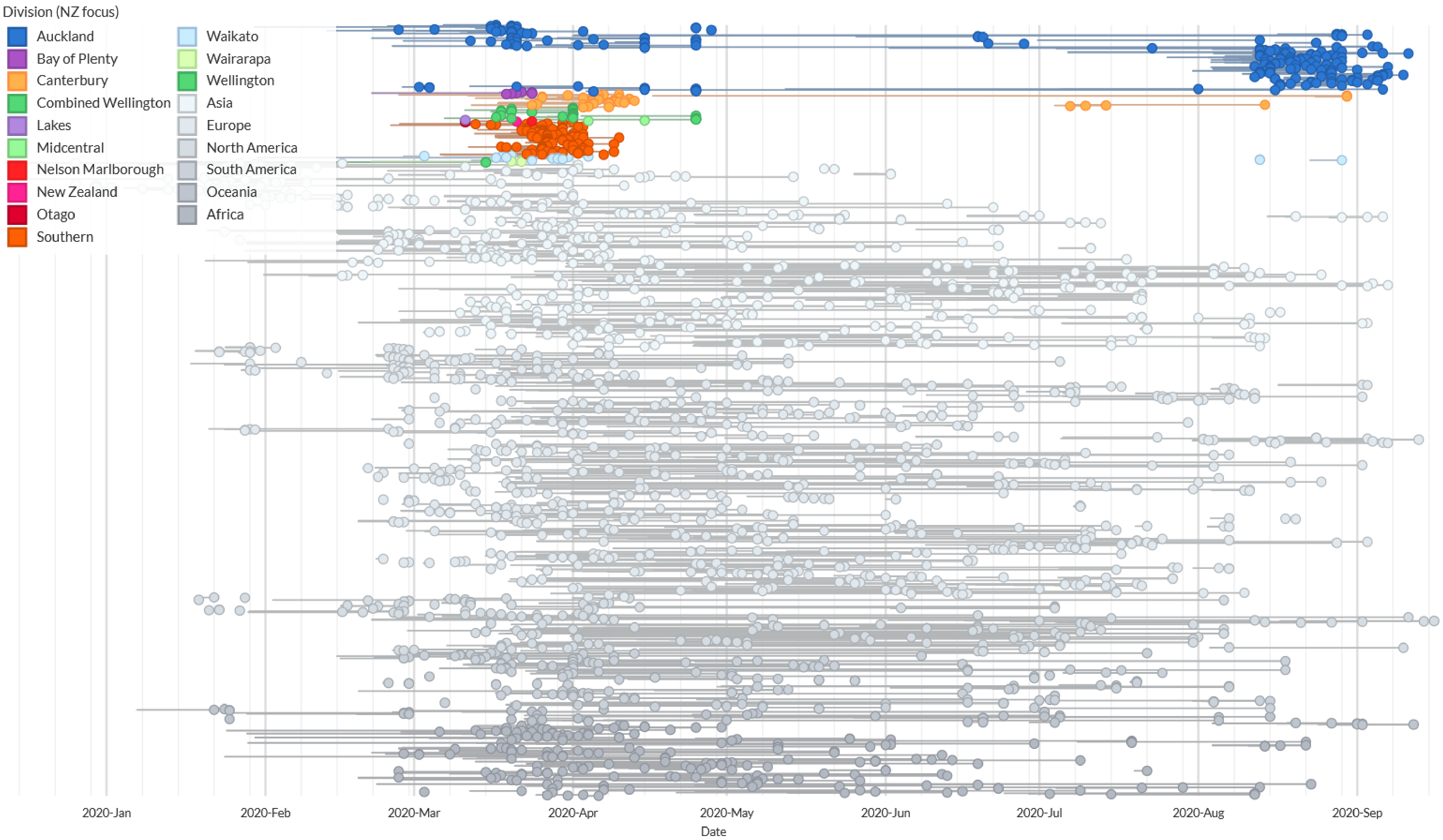
Motivation



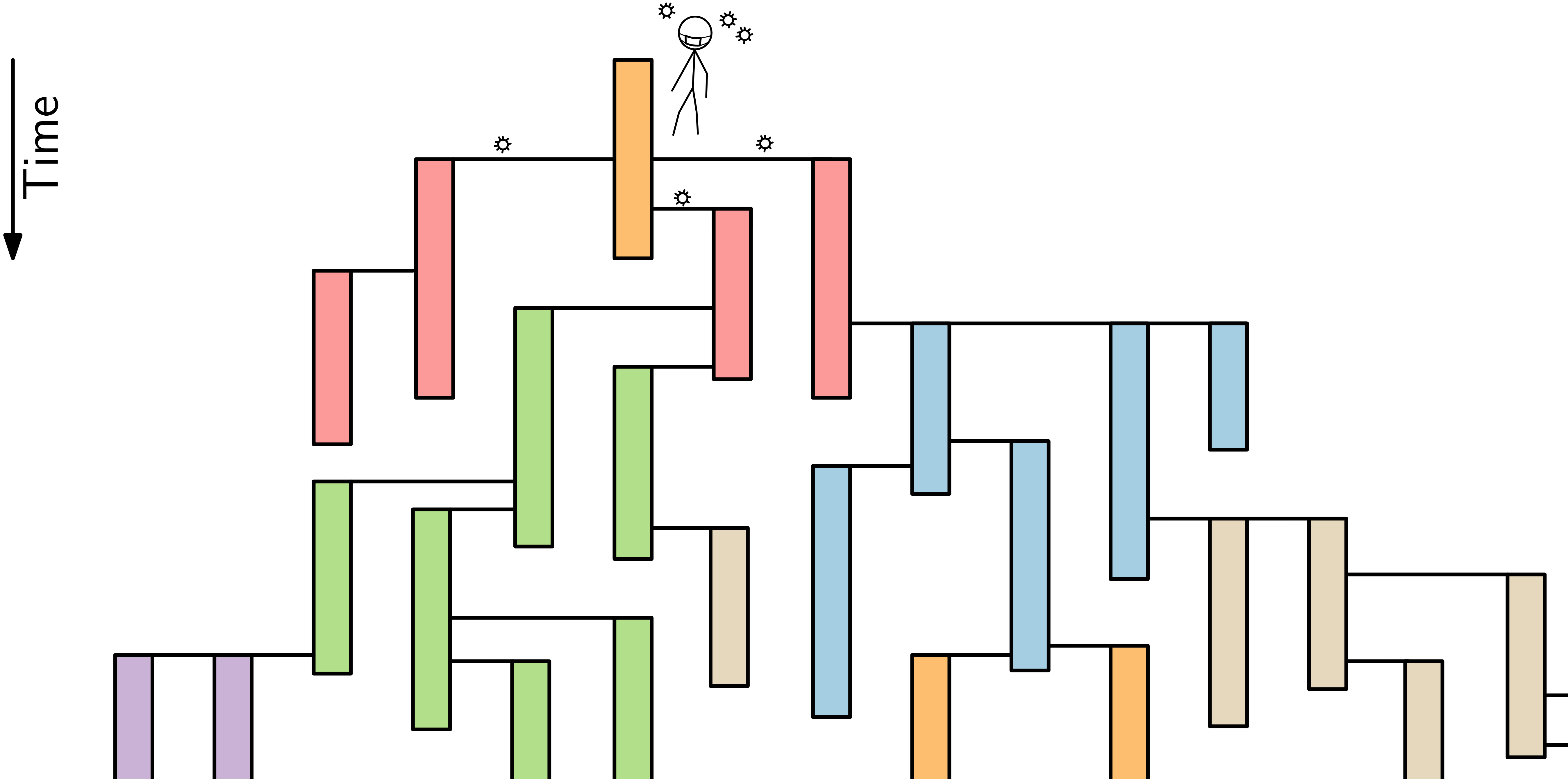
Motivation


Division (NZ focus)
Auckland
Bay of Plenty
Canterbury
Combined Wellington
Lakes
Midcentral
Nelson Marlborough
New Zealand
Otago
Southern
Waikato
Wairarapa
Wellington
Asia
Europe
North America
South America
Oceania
Africa
2020-Jan
2020-Feb
2020-Mar
2020-Apr
2020-May
2020-Jun
2020-Jul
2020-Aug
2020-Sep
Date

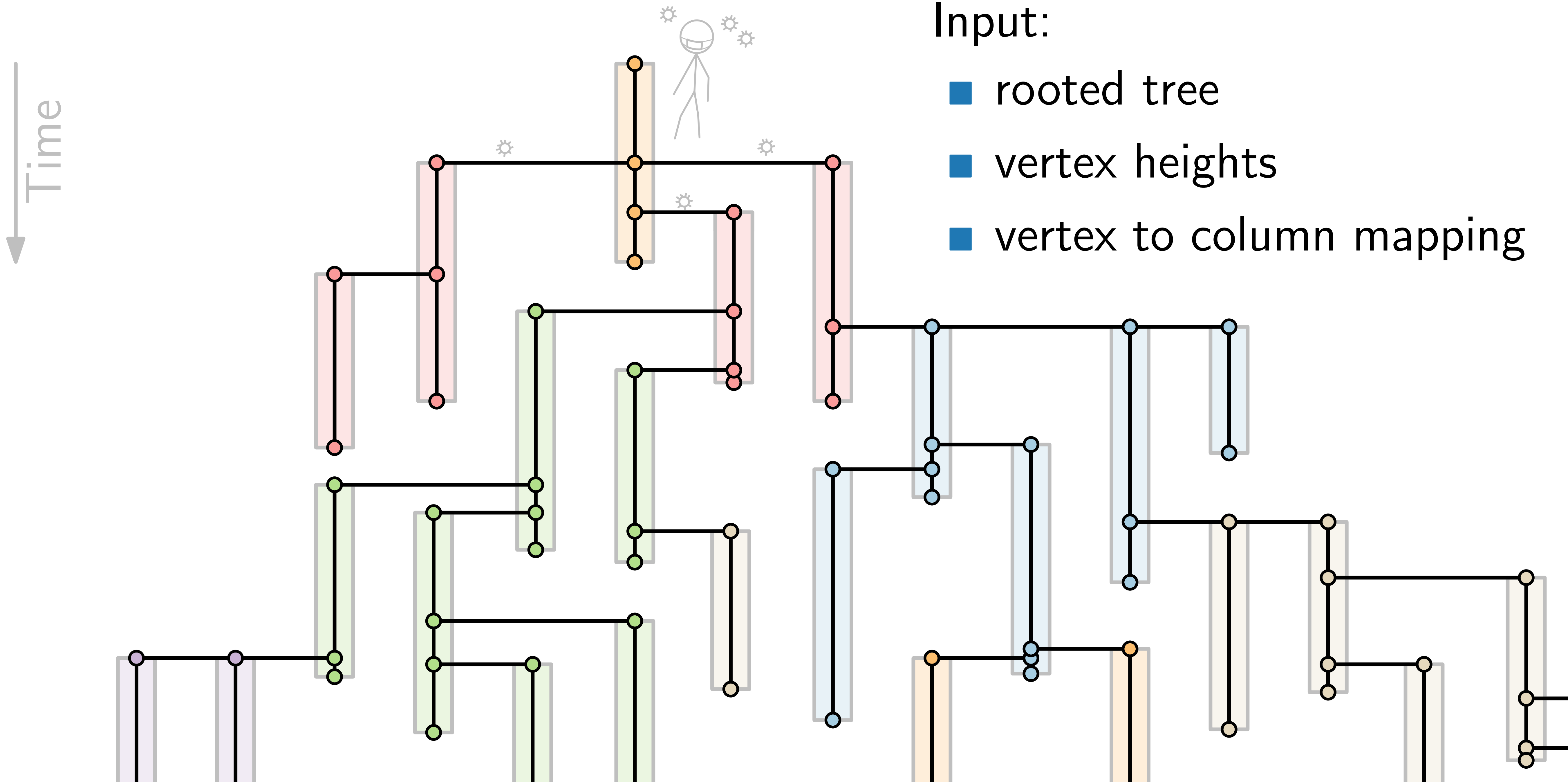
Motivation


Division (NZ focus)
Auckland
Bay of Plenty
Canterbury
Combined Wellington
Lakes
Midcentral
Nelson Marlborough
New Zealand
Otago
Southern
Waikato
Wairarapa
Wellington
Asia
Europe
North America
South America
Oceania
Africa
2020-Jan
2020-Feb
2020-Mar
2020-Apr
2020-May
2020-Jun
2020-Jul
2020-Aug
2020-Sep
Date

Motivation



Motivation



Input:

- rooted tree
- vertex heights
- vertex to column mapping

Overview

Drawing Style

Problem

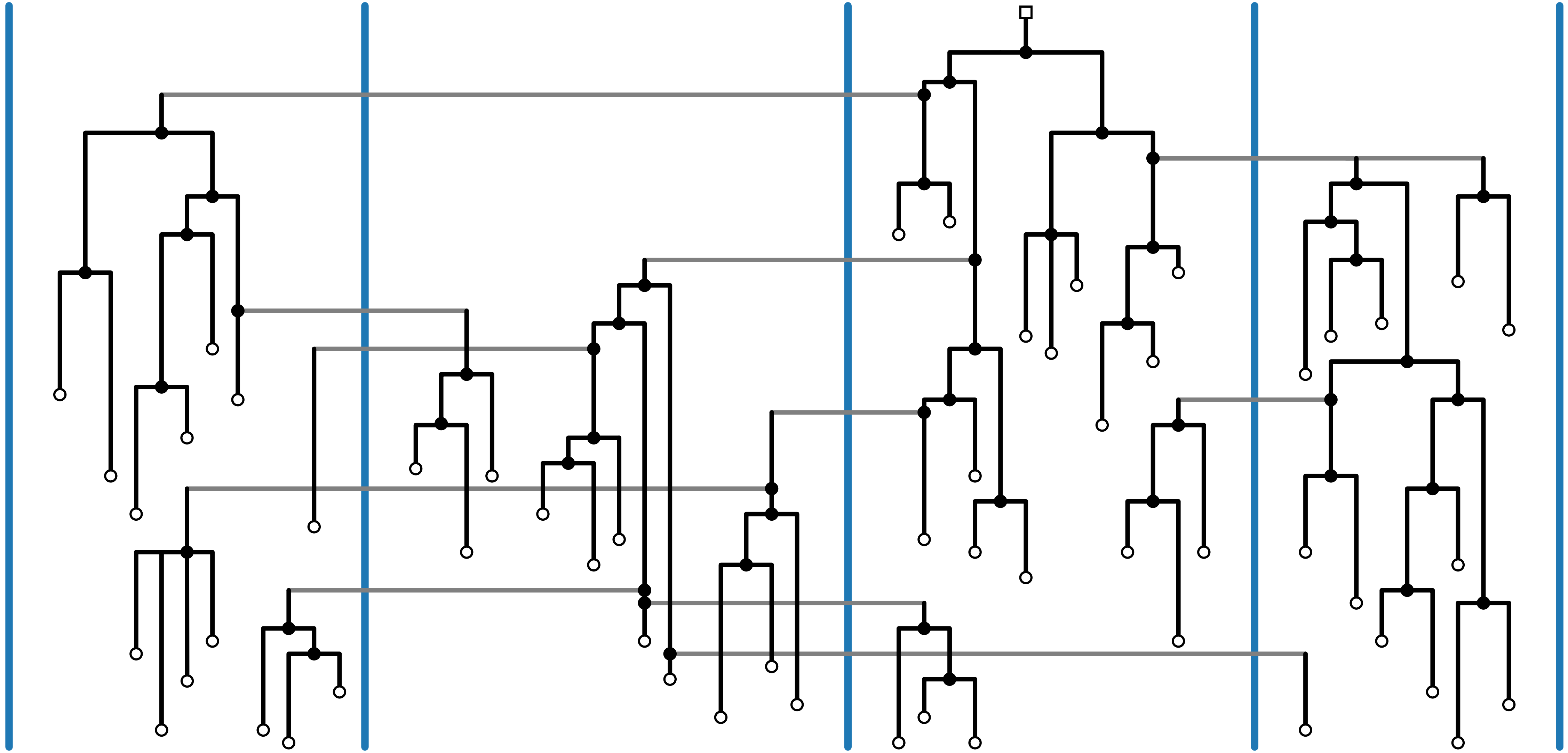
P

NPh

FPT

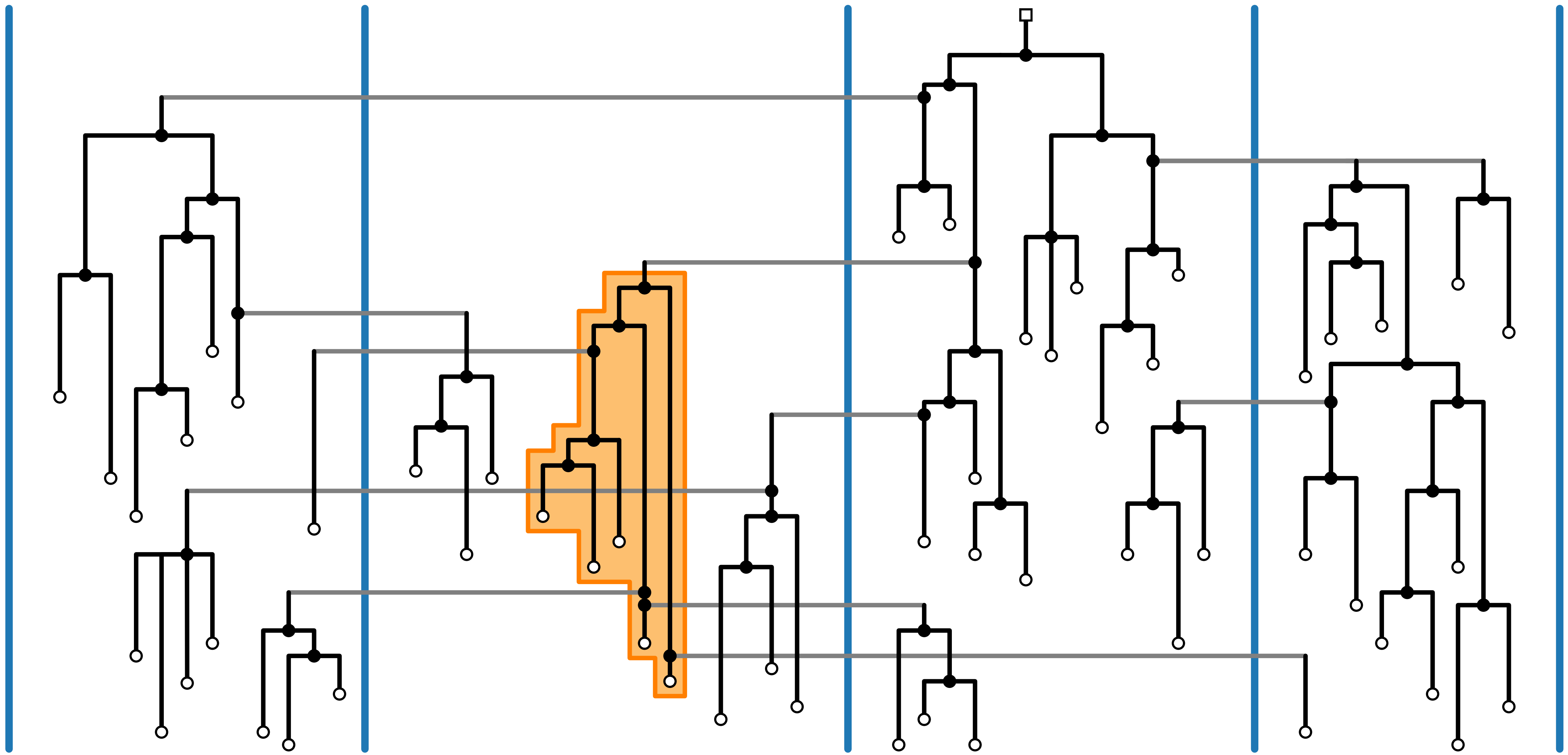
Drawing Style

- rectangular cladogram
- vertices in their columns at their heights



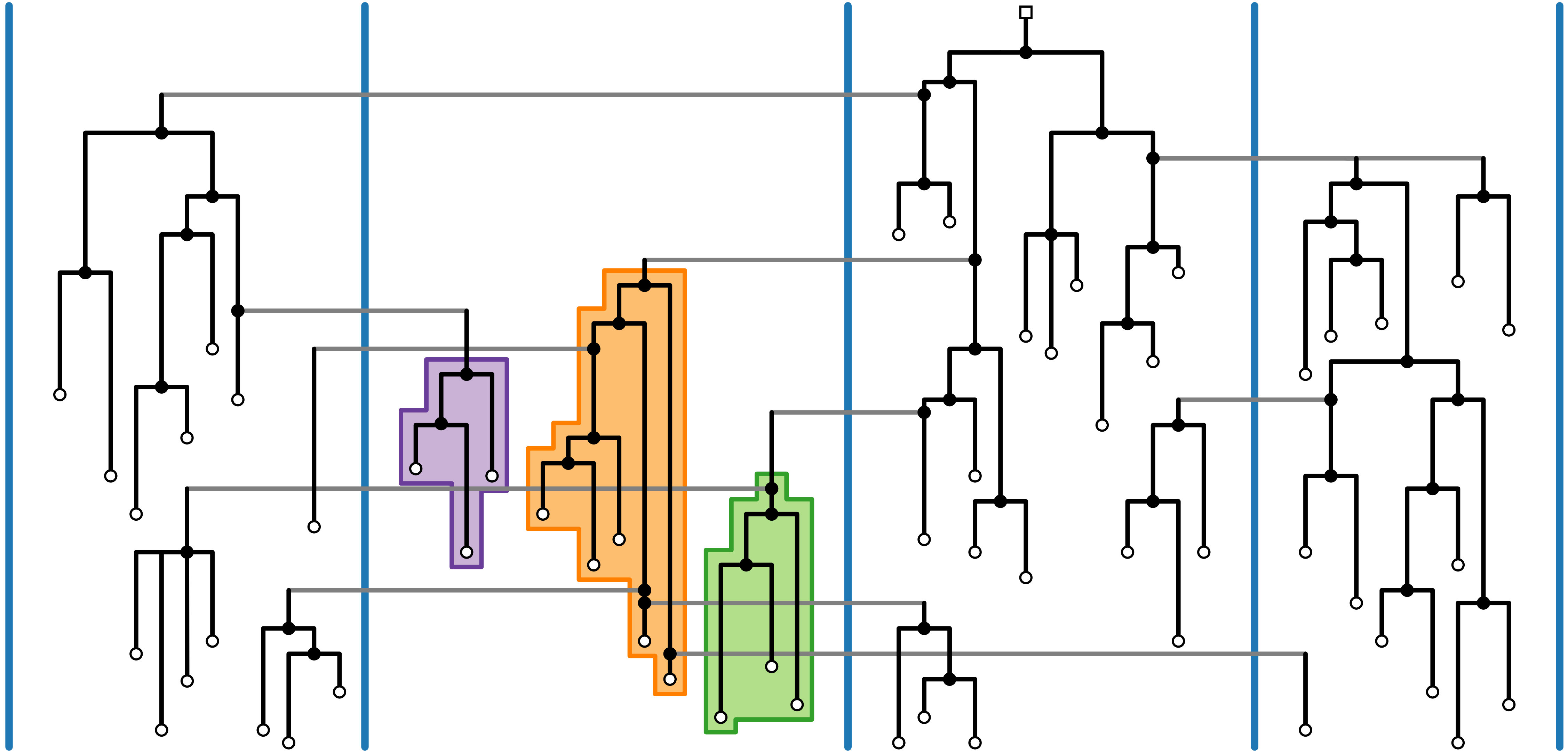
Drawing Style

- rectangular cladogram
- vertices in their columns at their heights



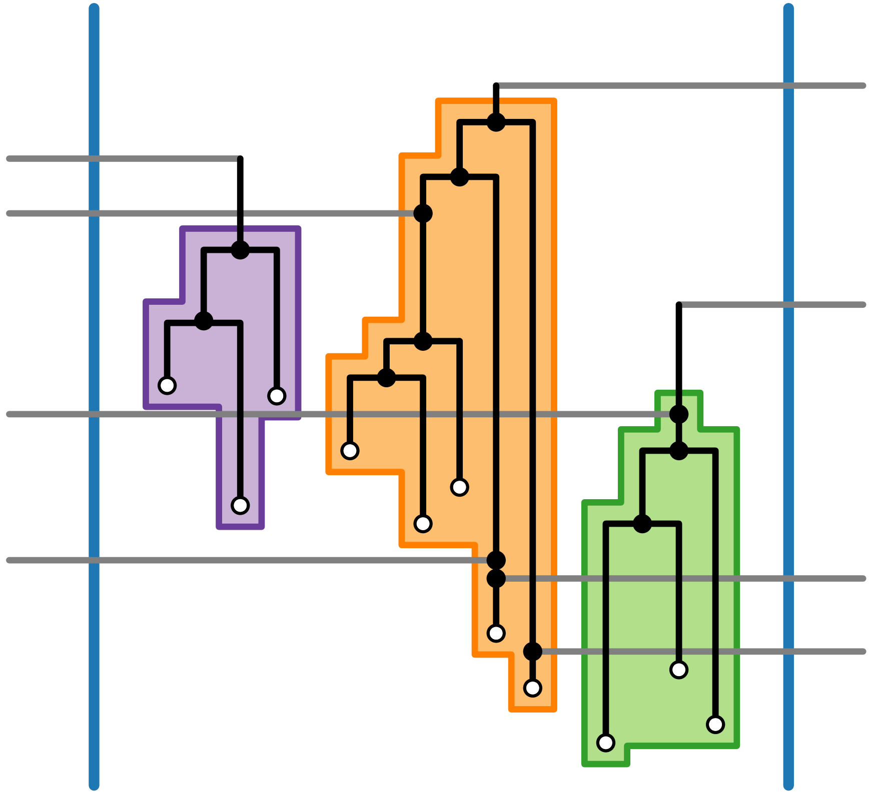
Drawing Style

- rectangular cladogram
- vertices in their columns at their heights



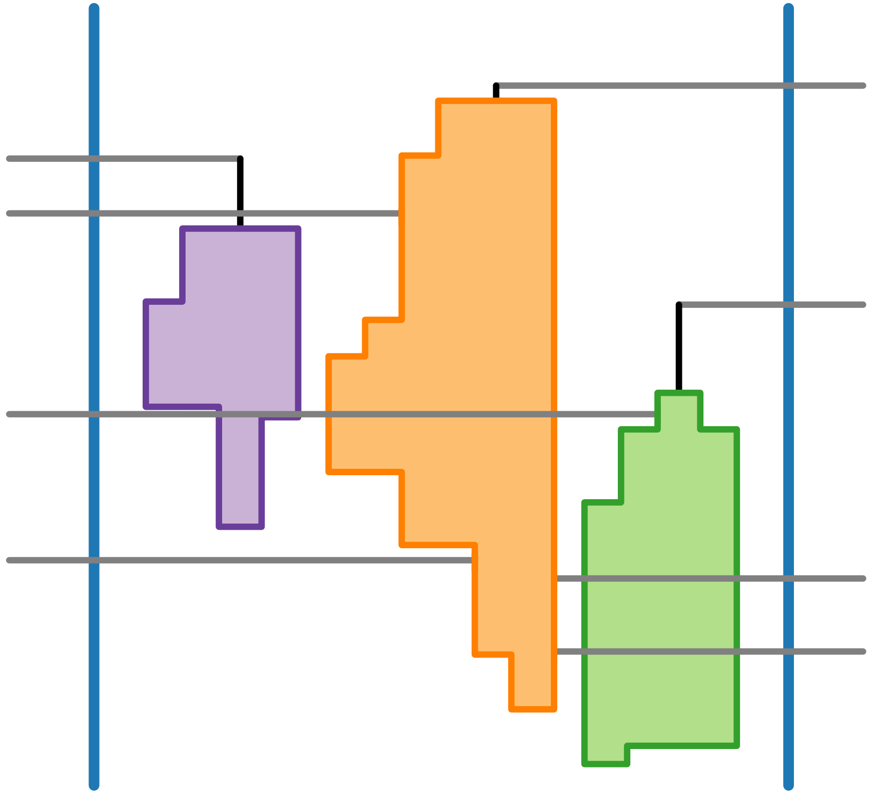
Drawing Style + Variants

- rectangular cladogram
- no two intra-column edges cross
- vertices in their columns at their heights



Drawing Style + Variants

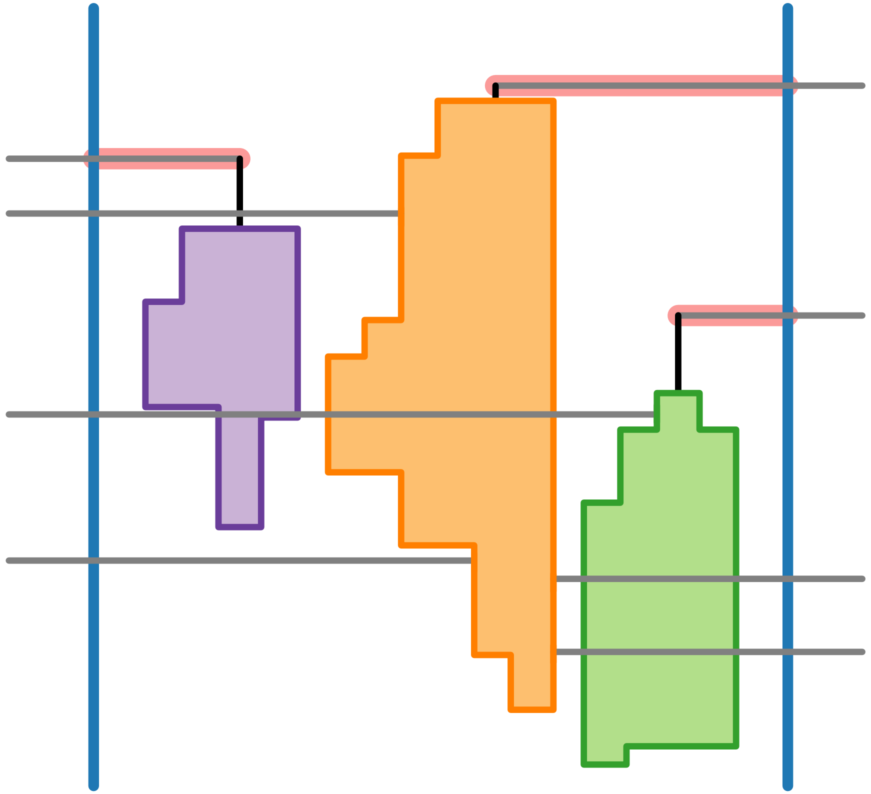
- rectangular cladogram
- vertices in their columns at their heights
- no two intra-column edges cross



Drawing Style + Variants

- rectangular cladogram
- no two intra-column edges cross
- vertices in their columns at their heights

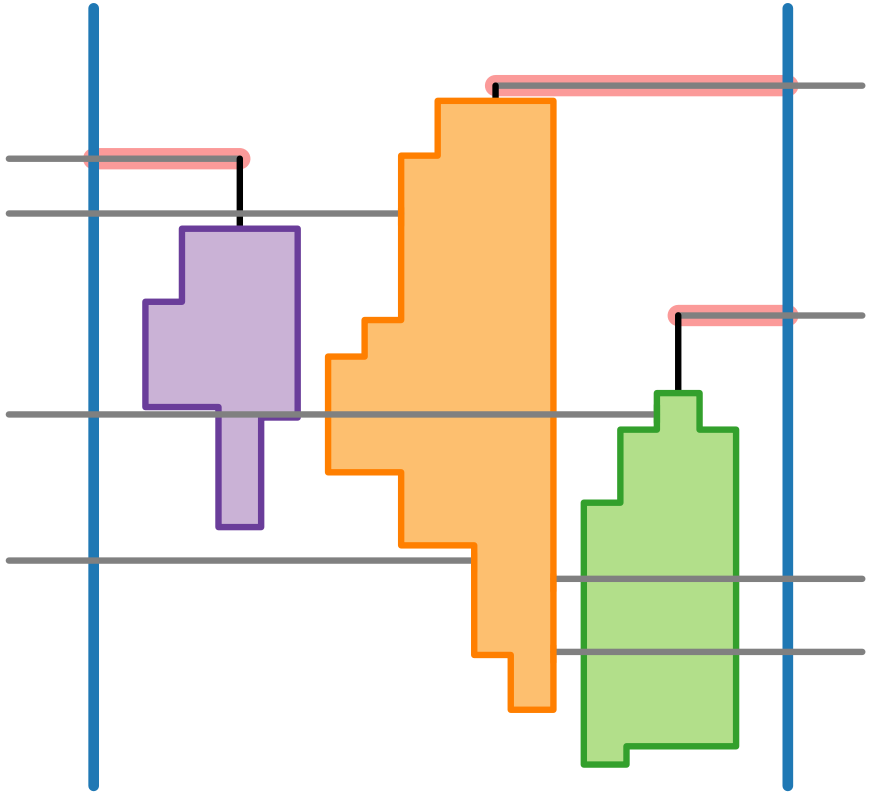
“directly at column border”



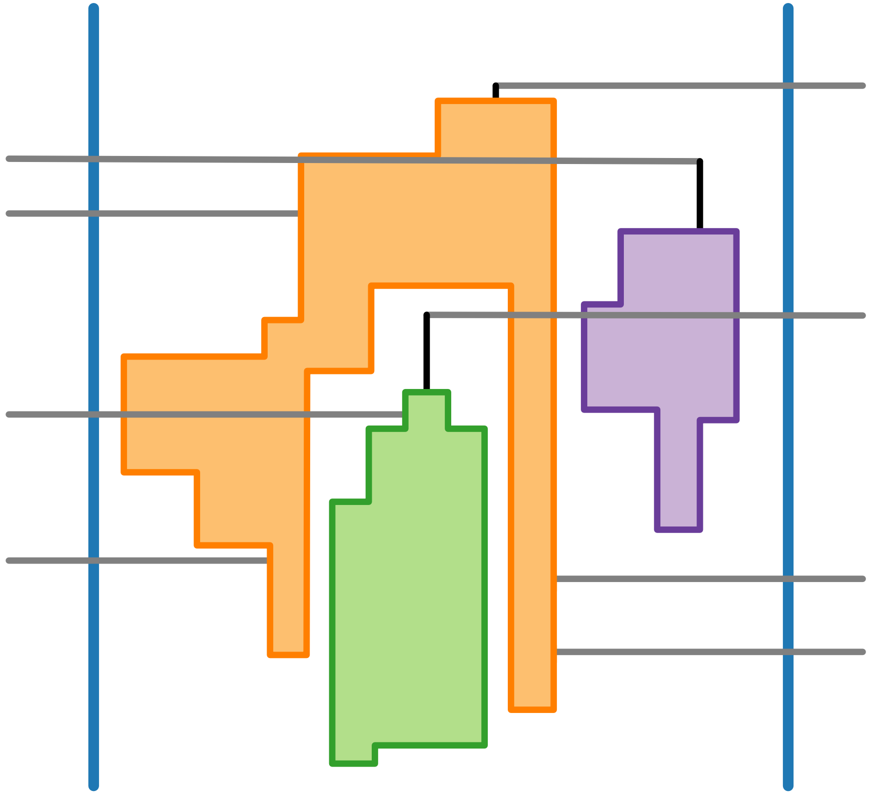
Drawing Style + Variants

- rectangular cladogram
- vertices in their columns at their heights
- no two intra-column edges cross

“directly at column border”

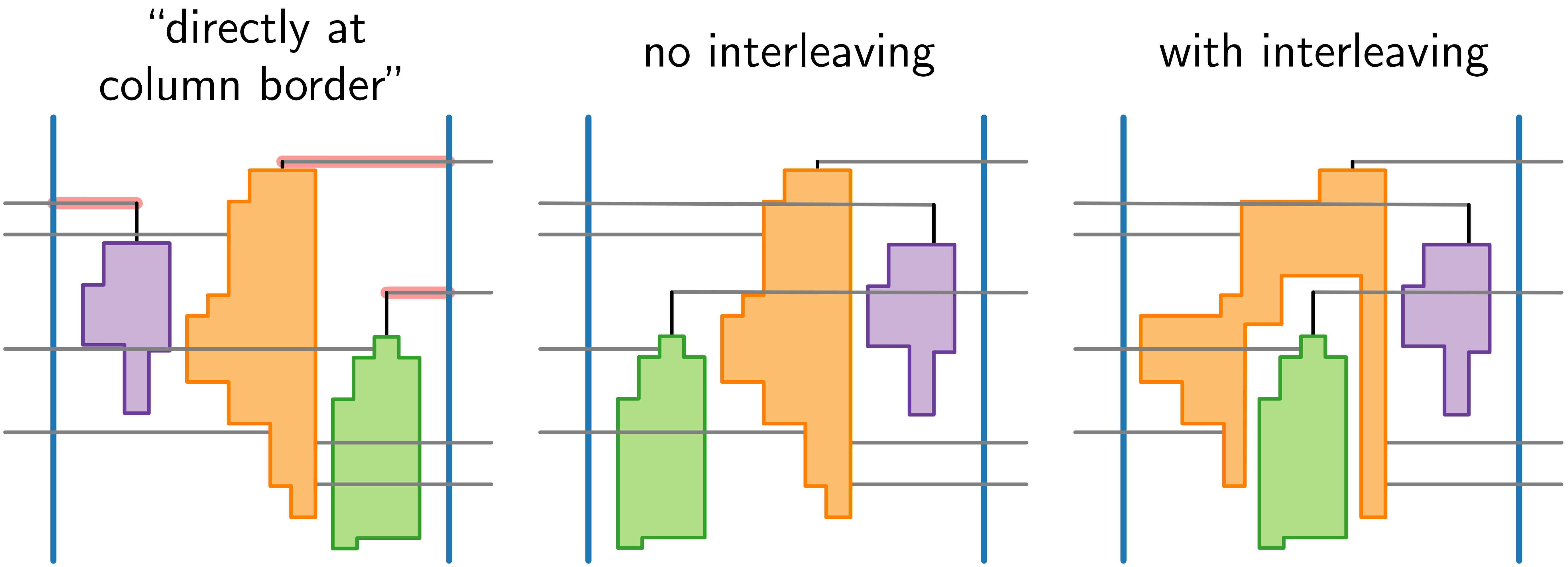


with interleaving



Drawing Style + Variants

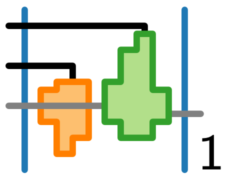
- rectangular cladogram
- vertices in their columns at their heights
- no two intra-column edges cross



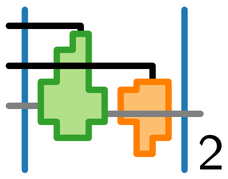
Drawing Style + Variants

- rectangular cladogram
- vertices in their columns at their heights
- no two intra-column edges cross

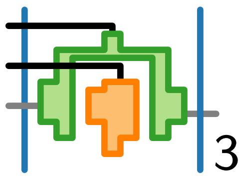
“directly at column border”



no interleaving



with interleaving



Crossing Minimisation Problem

Find a column tree embedding
with min number of crossings.

Crossing Minimisation Problem

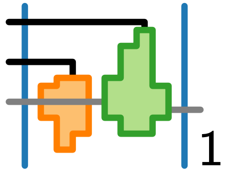
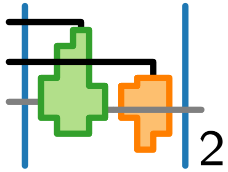
Find a column tree embedding
with min number of crossings.

- need to find **subtree embedding**
- need to find **subtree arrangement**

Crossing Minimisation Problem

Find a column tree embedding
with min number of crossings.

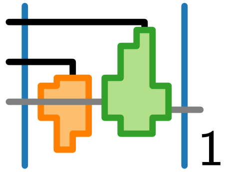
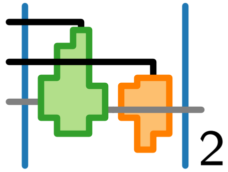
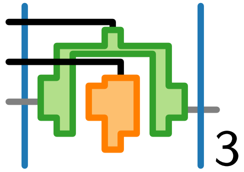
- need to find **subtree embedding**
- need to find **subtree arrangement**

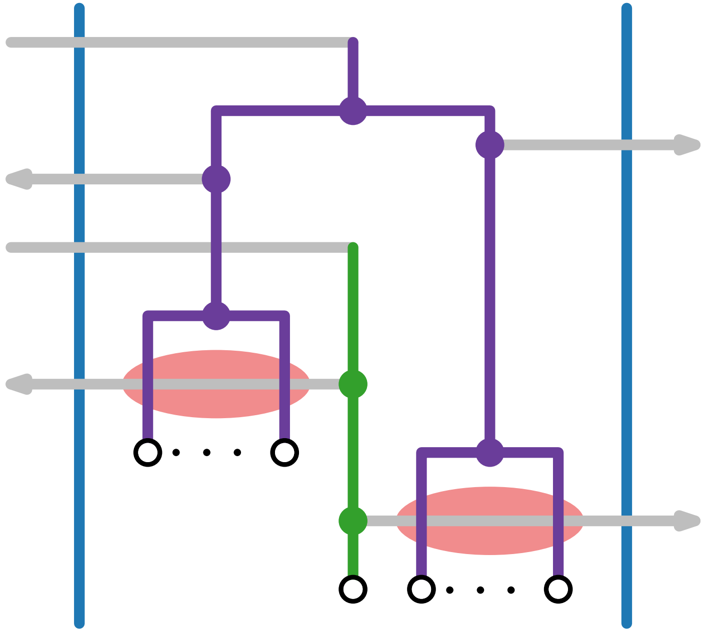
- independent tasks for  and 

Crossing Minimisation Problem

Find a column tree embedding
with min number of crossings.

- need to find **subtree embedding**
- need to find **subtree arrangement**

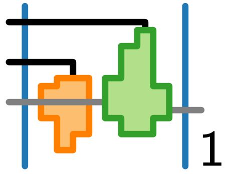
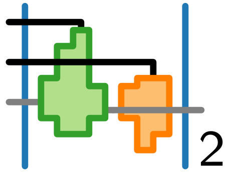
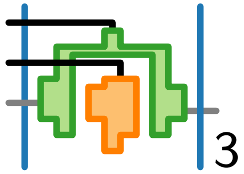
- independent tasks for  1 and  2
- . . . but not so for  3

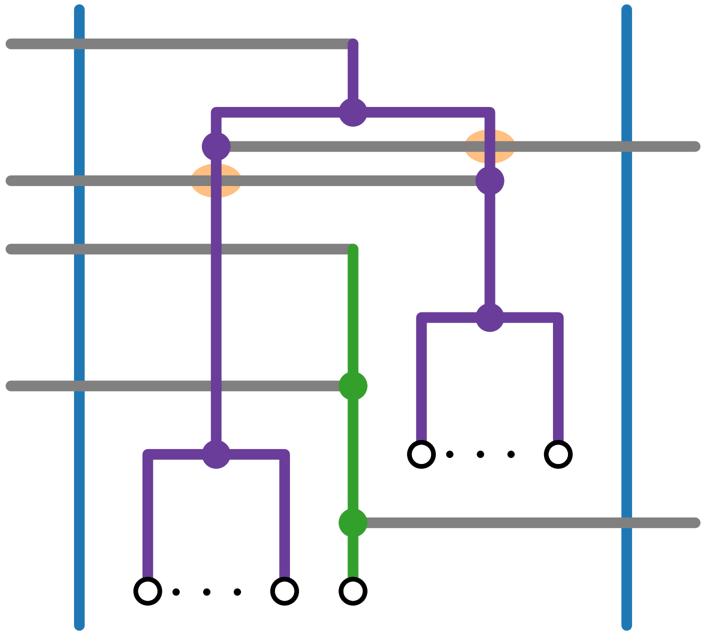


Crossing Minimisation Problem

Find a column tree embedding
with min number of crossings.

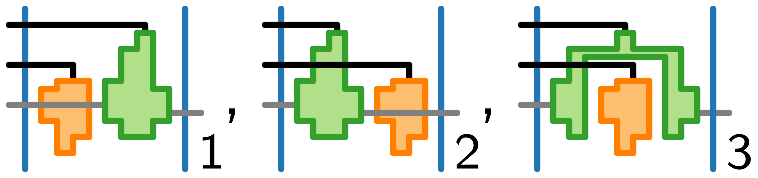
- need to find **subtree embedding**
- need to find **subtree arrangement**

- independent tasks for  and 
- . . . but not so for 

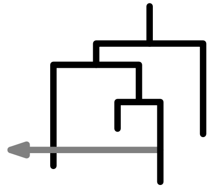


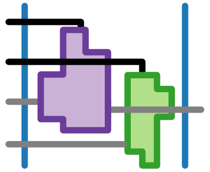
Overview

Drawing Style



Crossing Minimisation



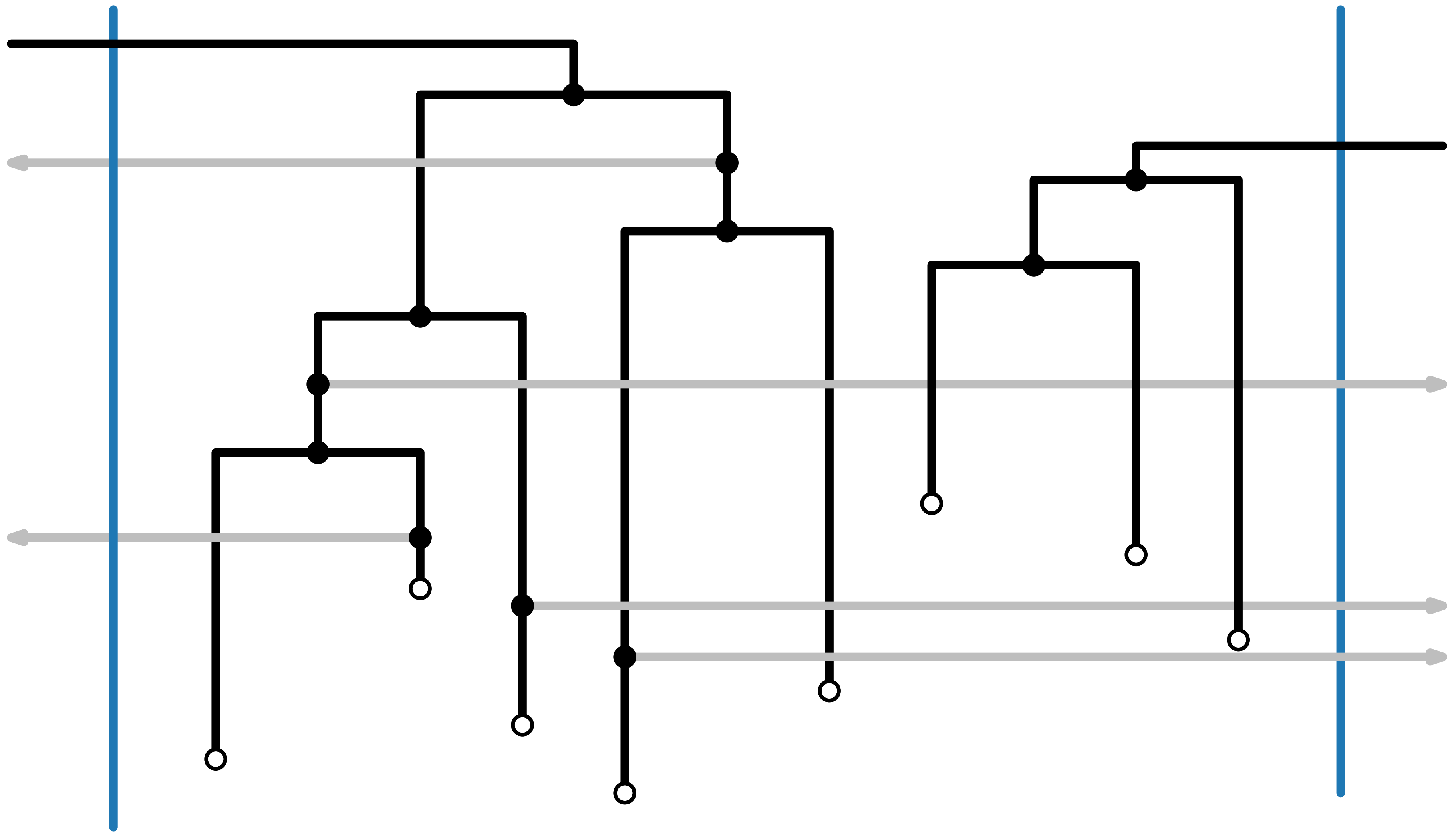


P

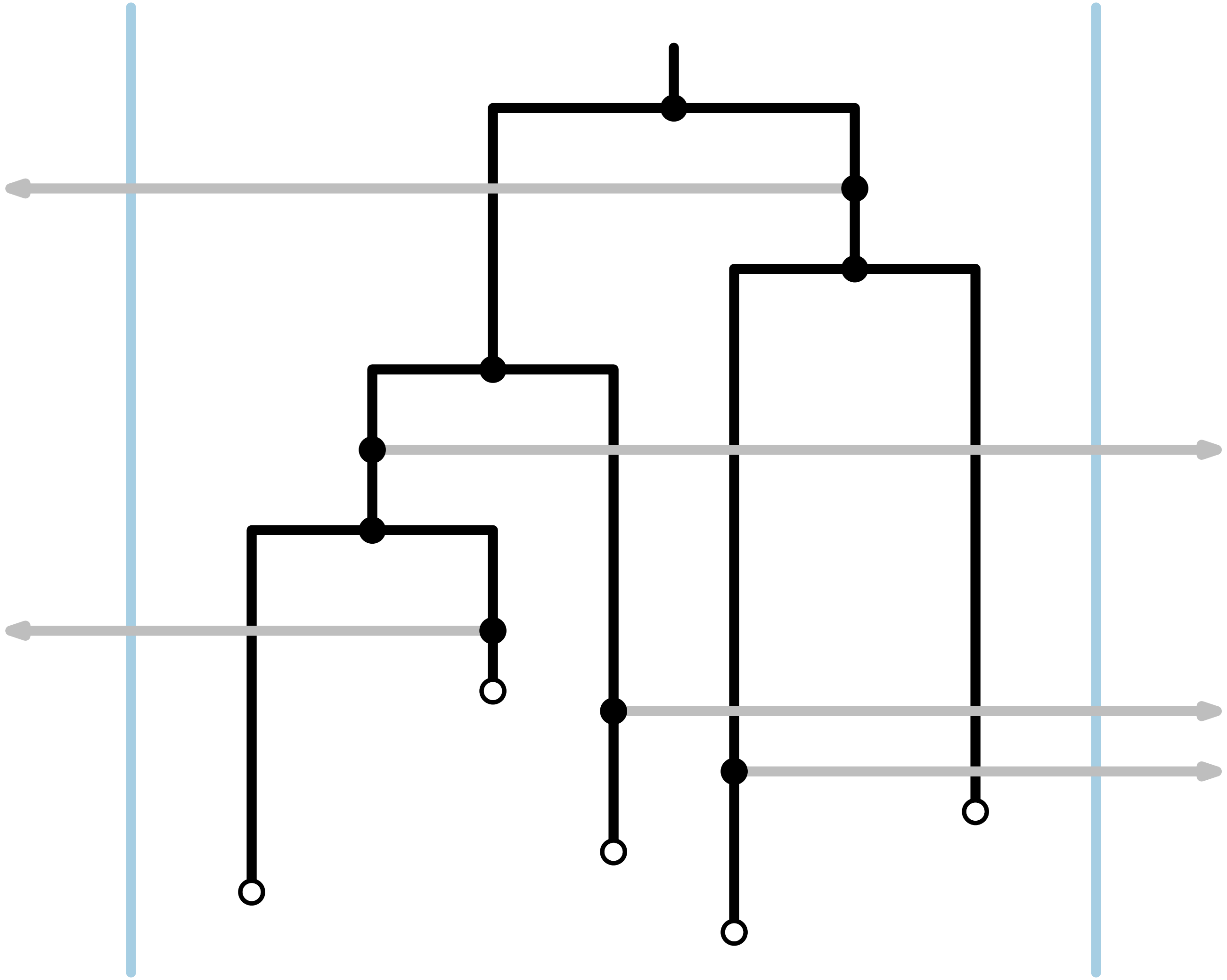
NP

FPT

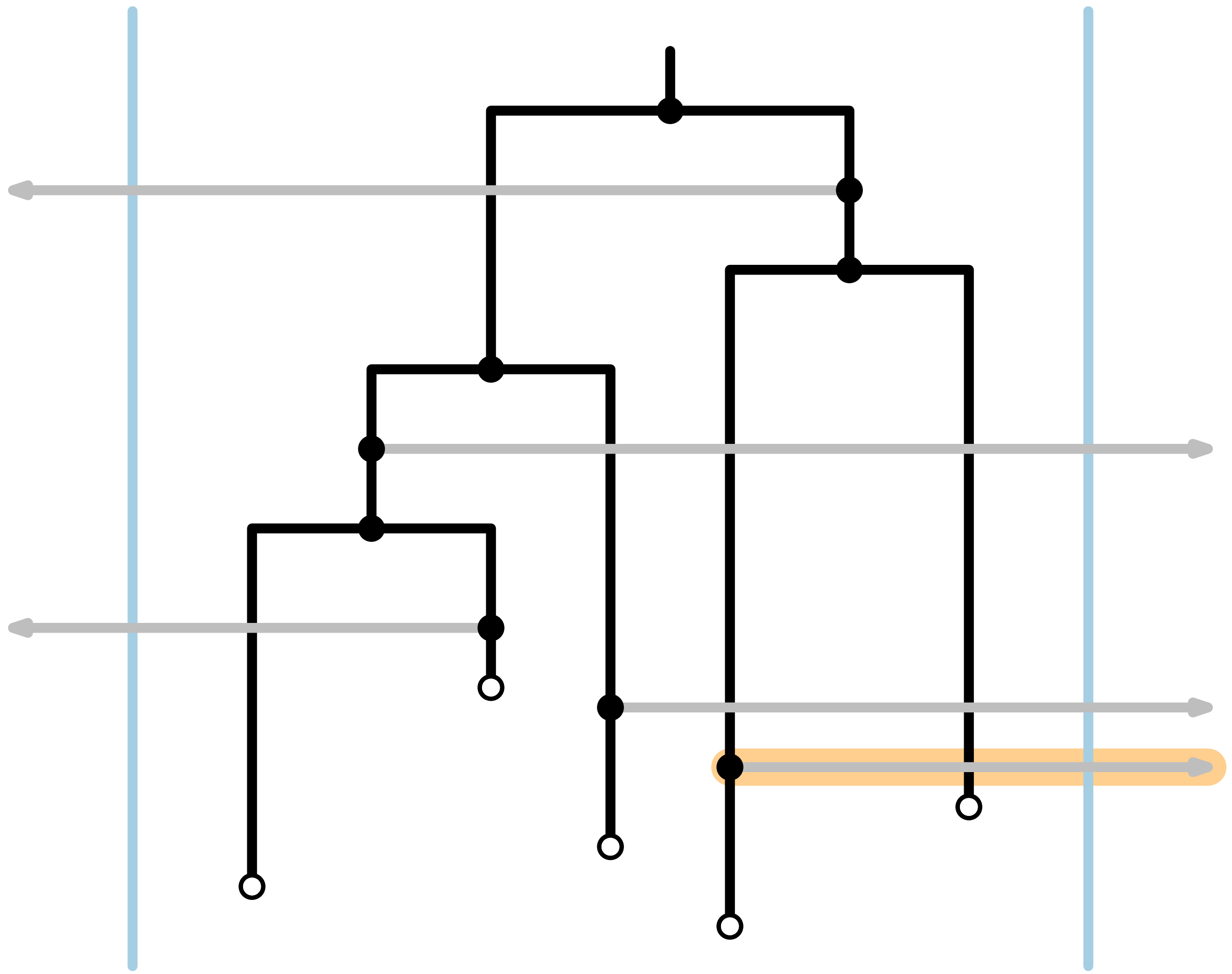
Subtree Embedding Algorithm



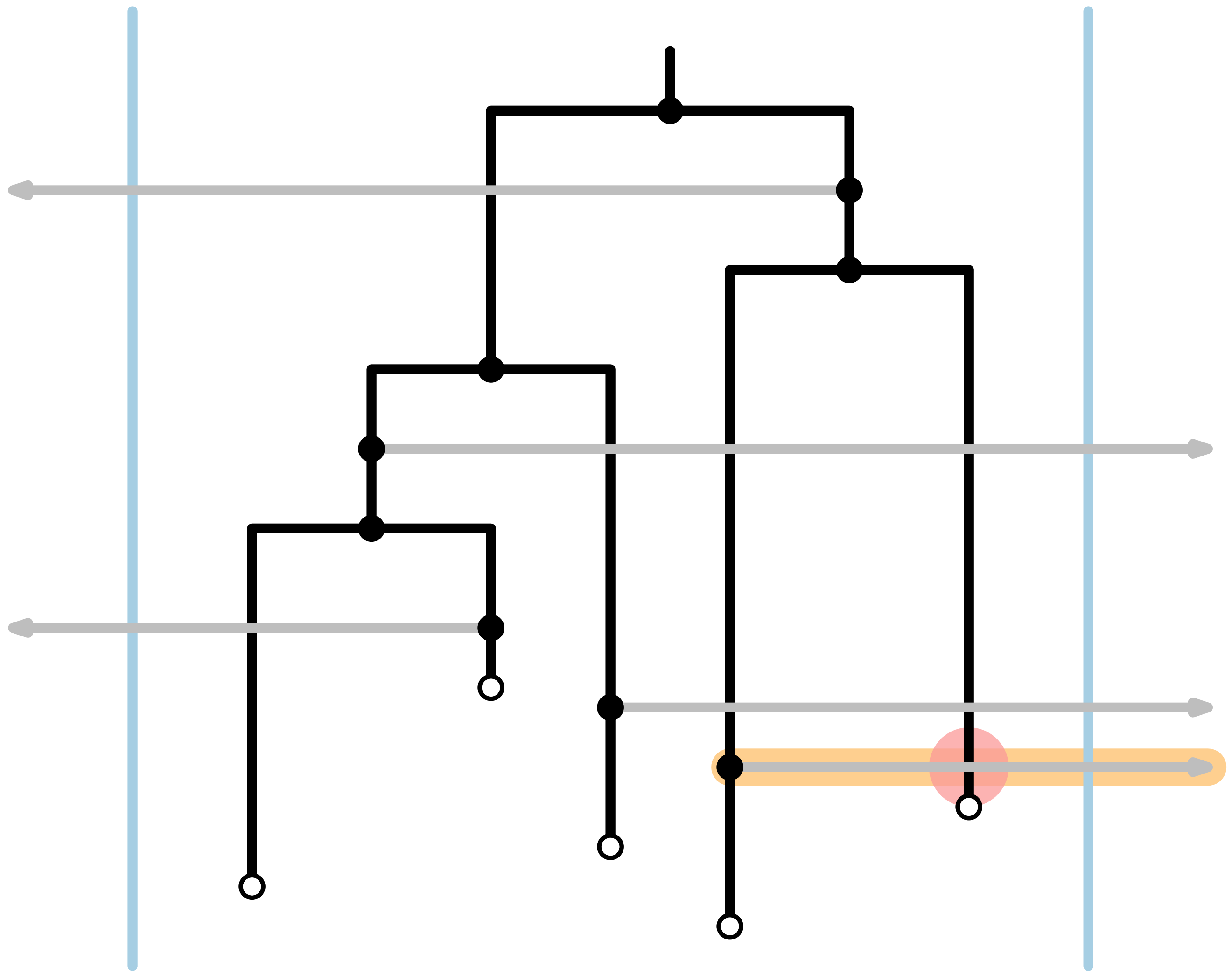
Subtree Embedding Algorithm



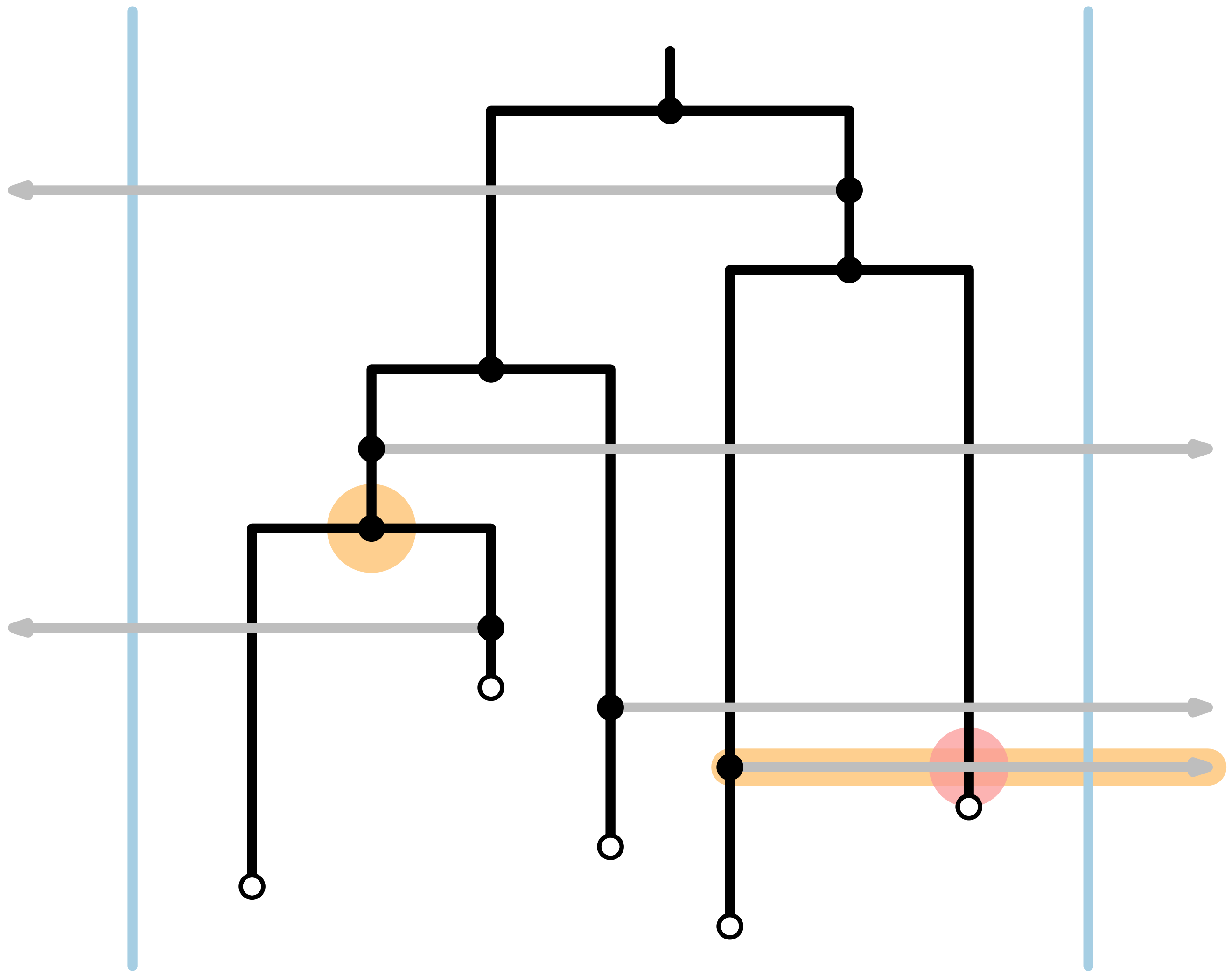
Subtree Embedding Algorithm



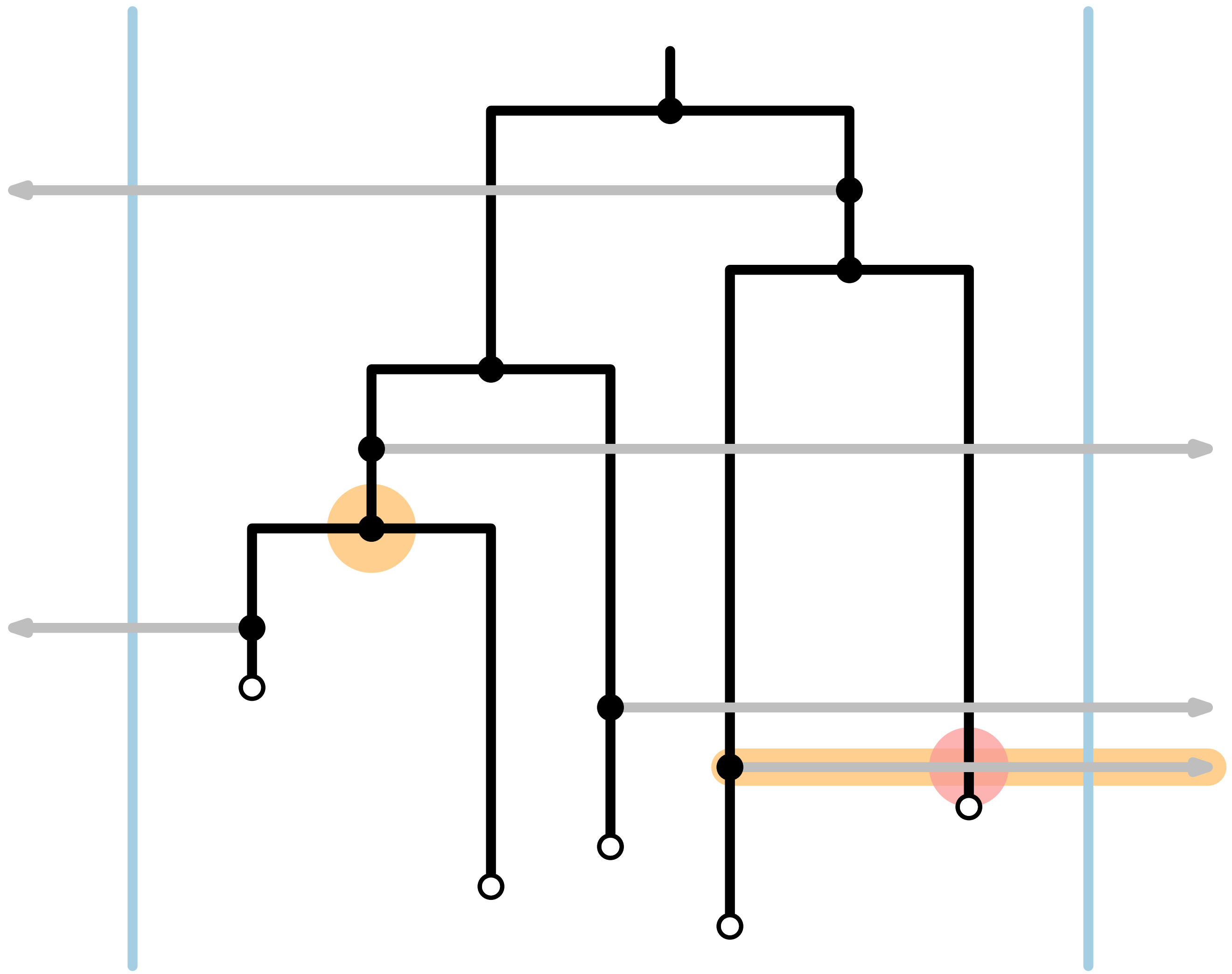
Subtree Embedding Algorithm



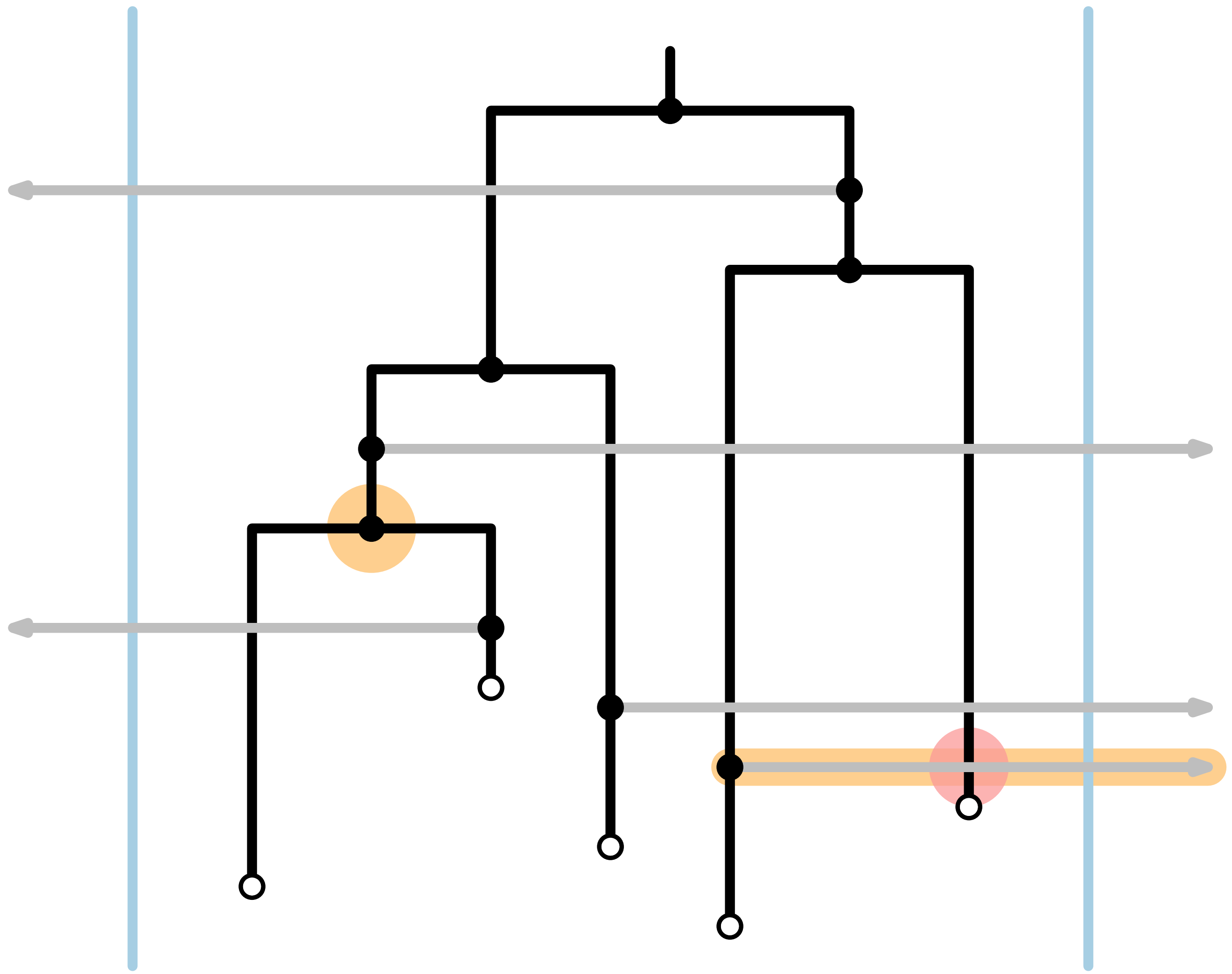
Subtree Embedding Algorithm



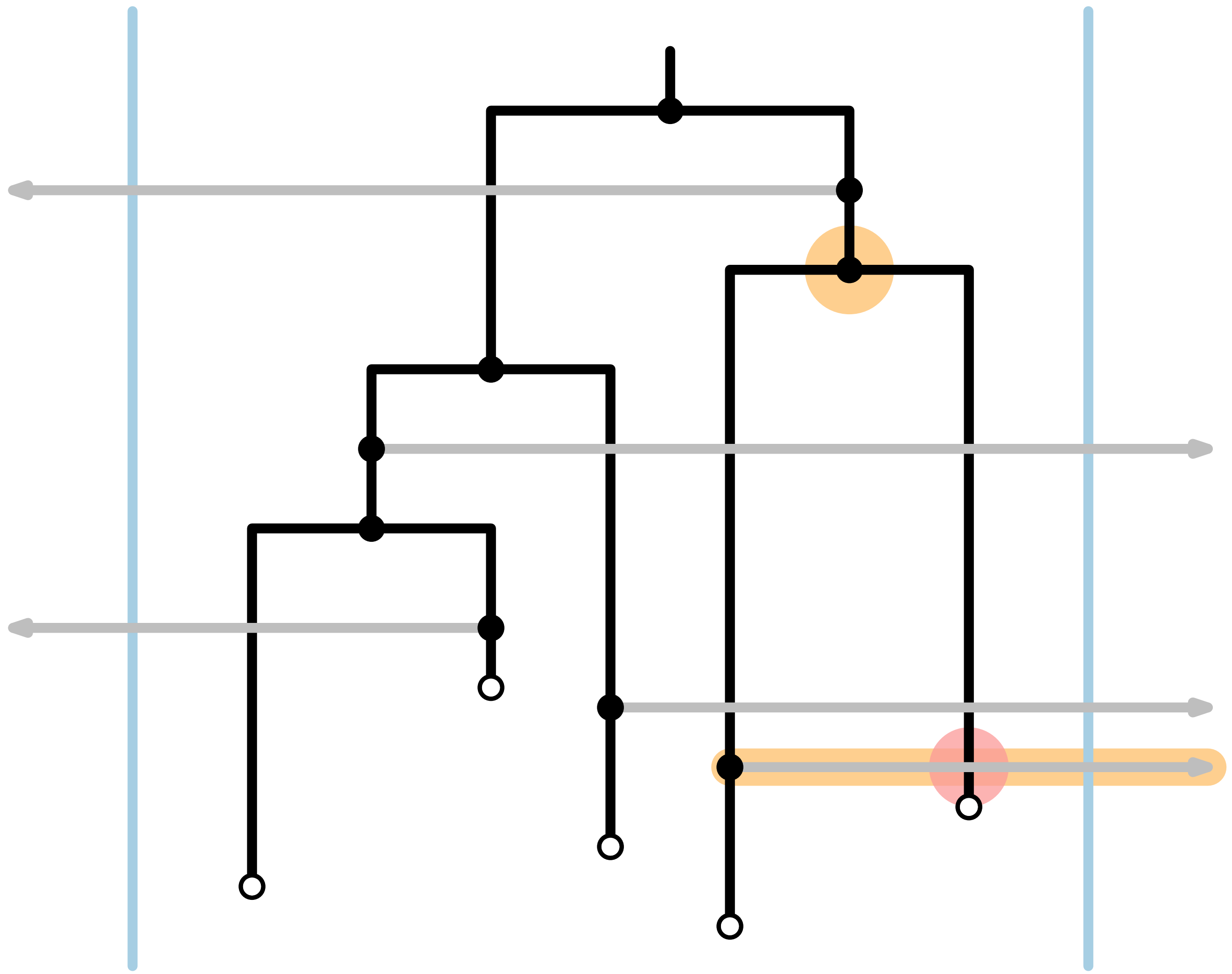
Subtree Embedding Algorithm



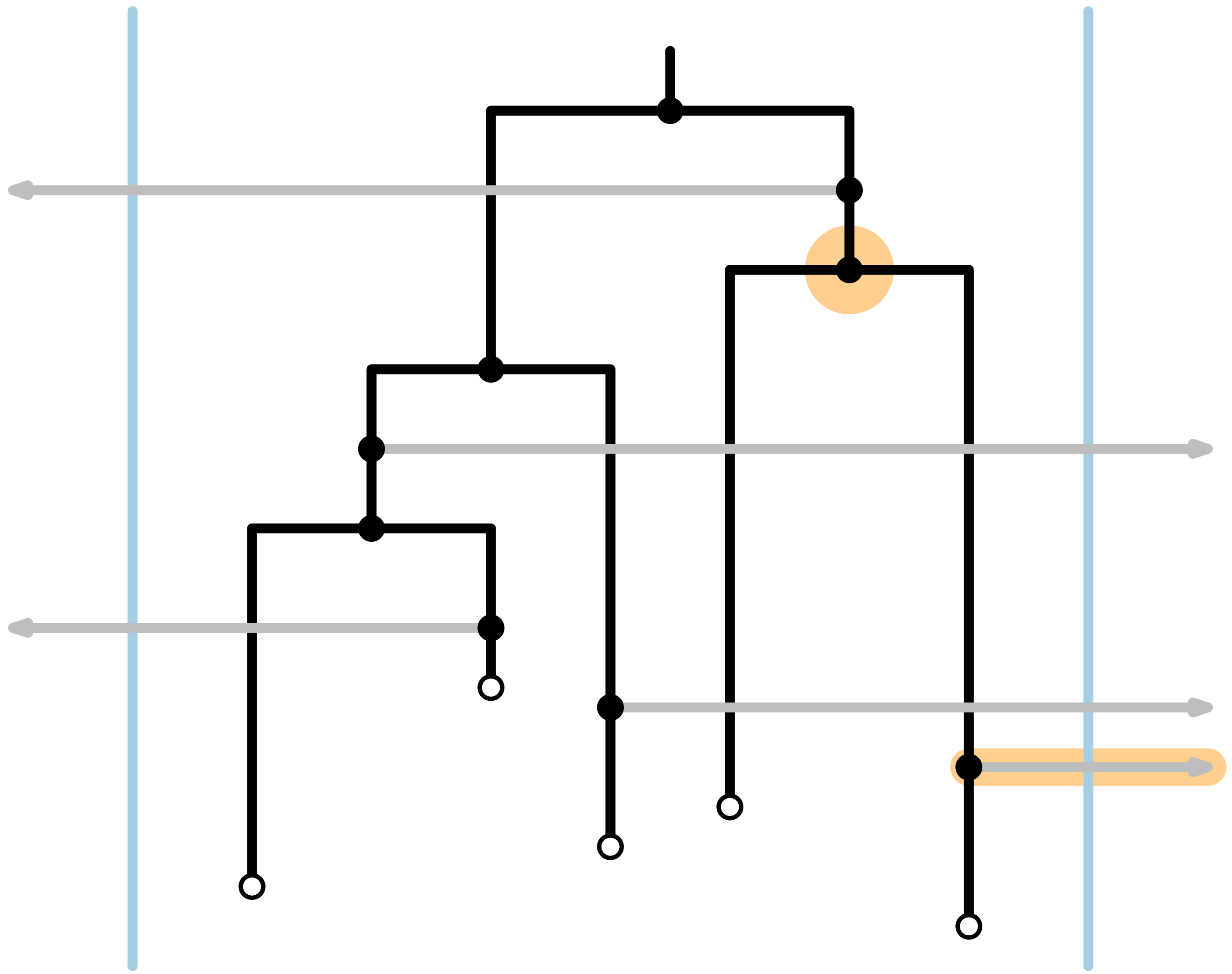
Subtree Embedding Algorithm



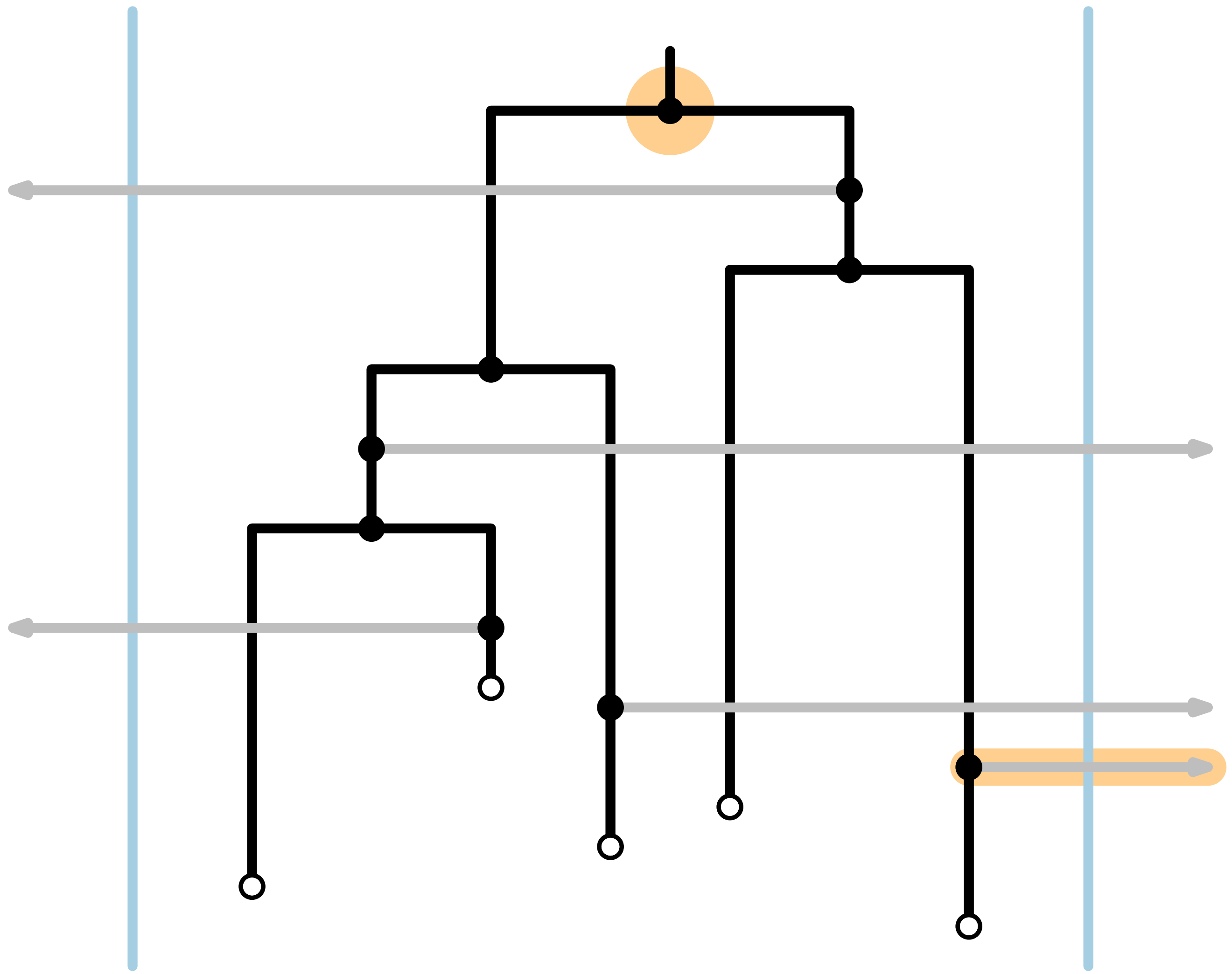
Subtree Embedding Algorithm



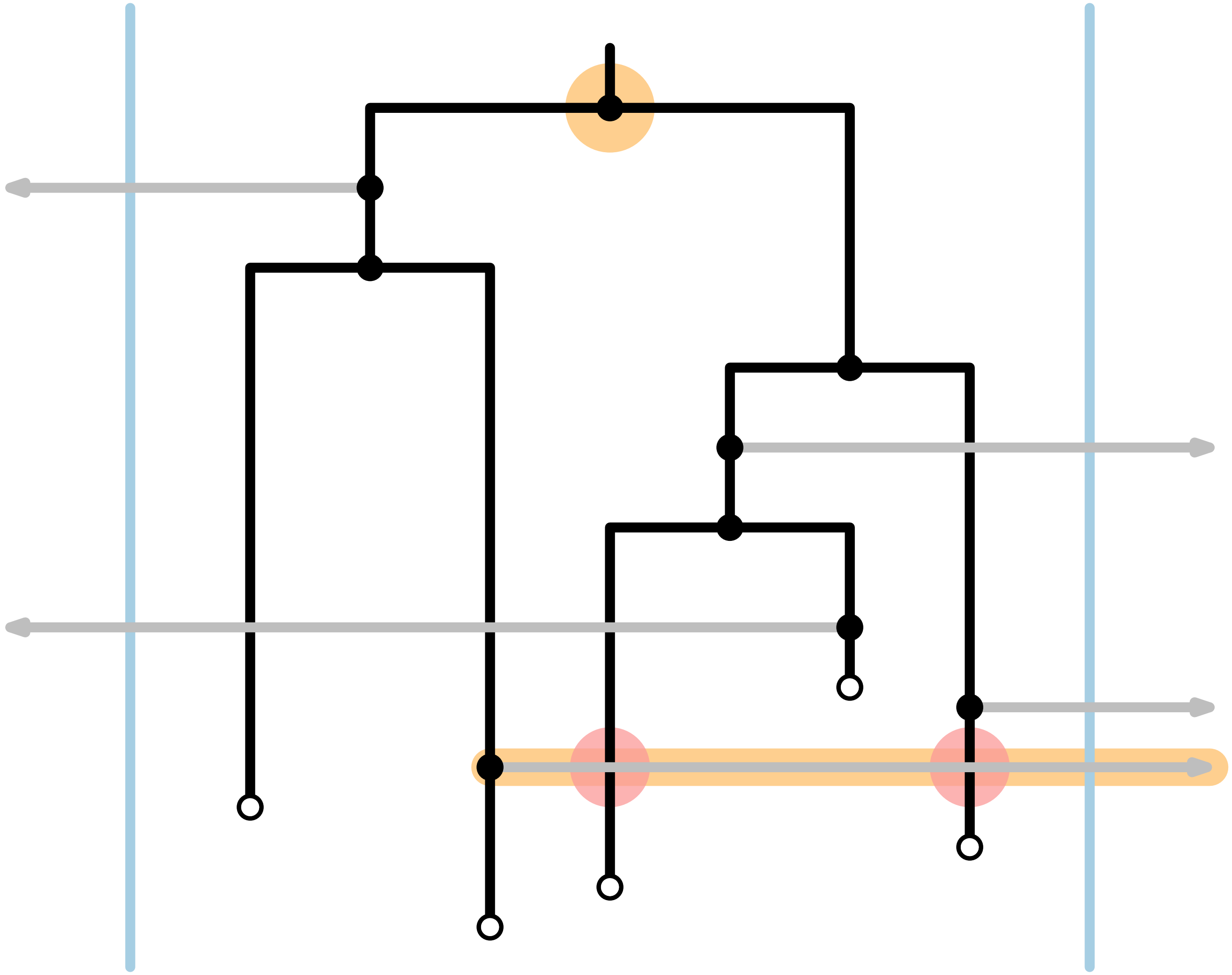
Subtree Embedding Algorithm



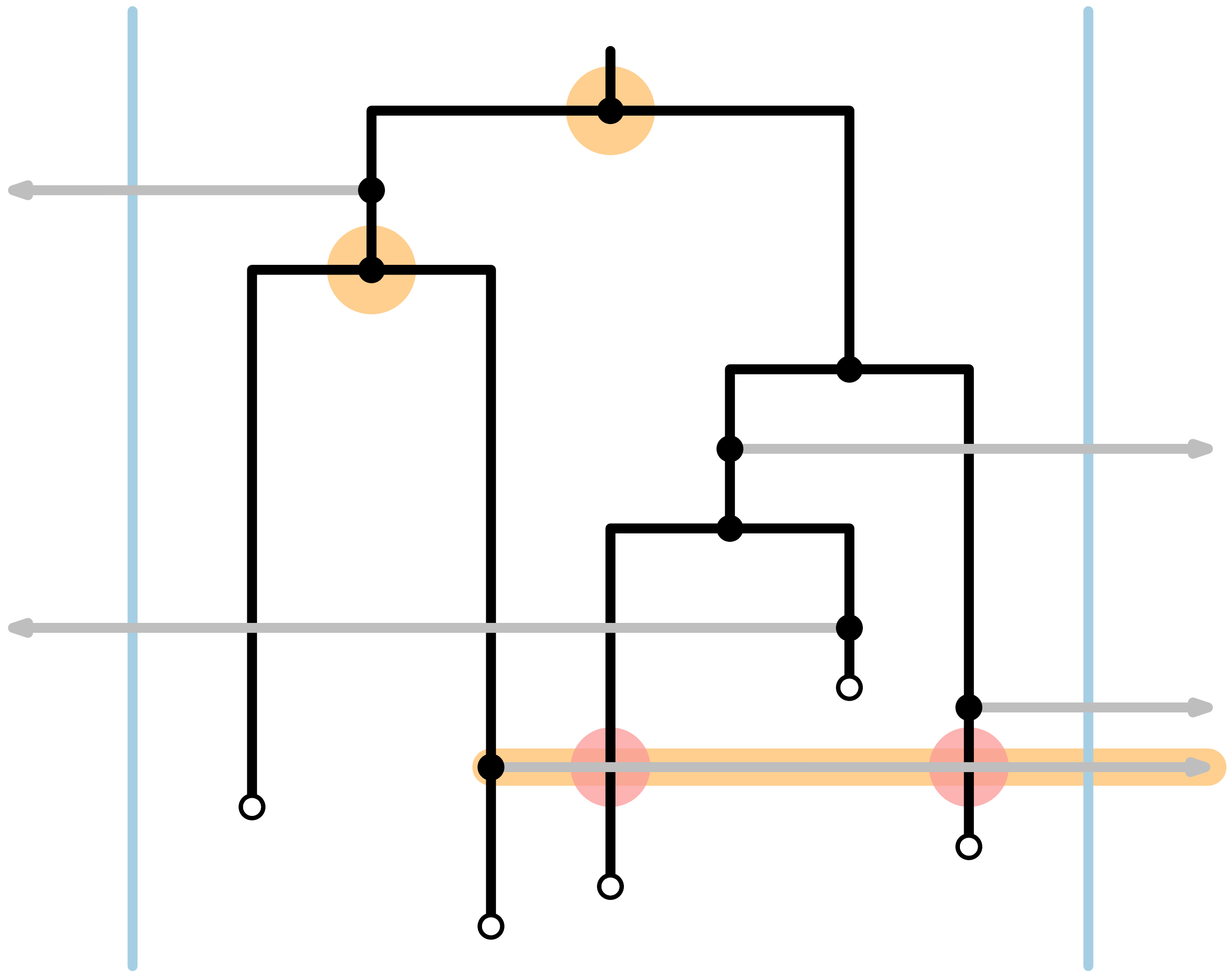
Subtree Embedding Algorithm



Subtree Embedding Algorithm

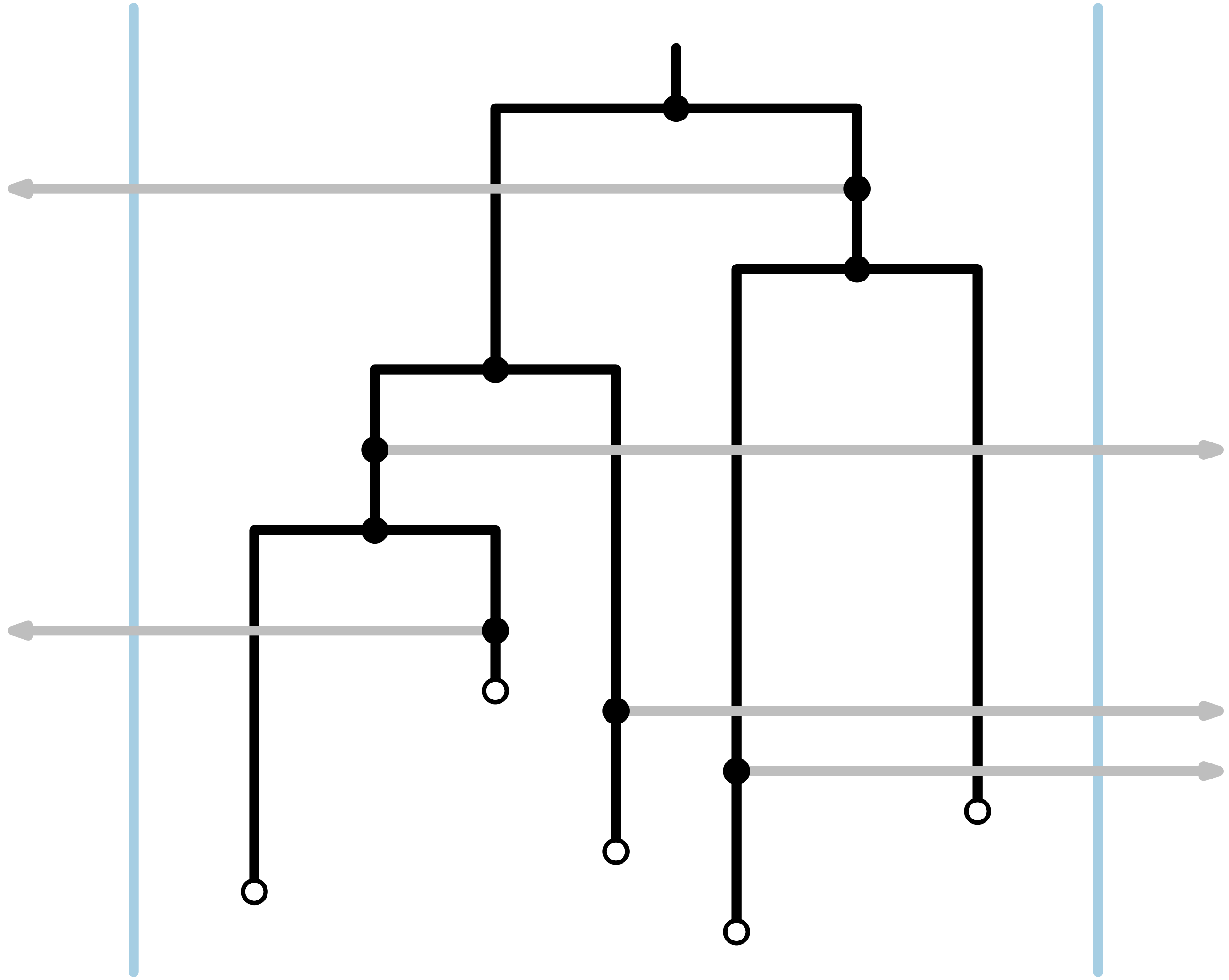


Subtree Embedding Algorithm



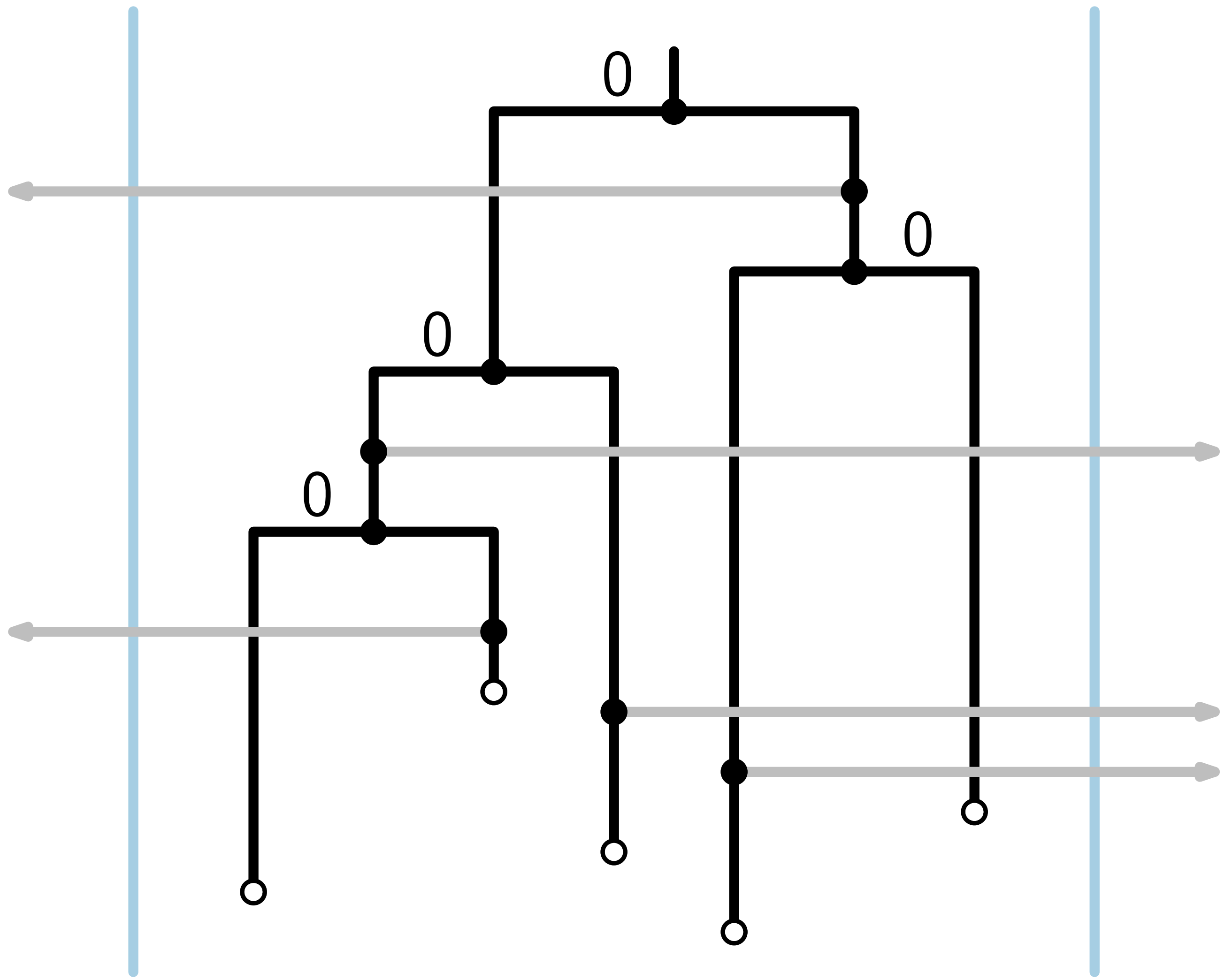
- only rotations of ancestors matter

Subtree Embedding Algorithm



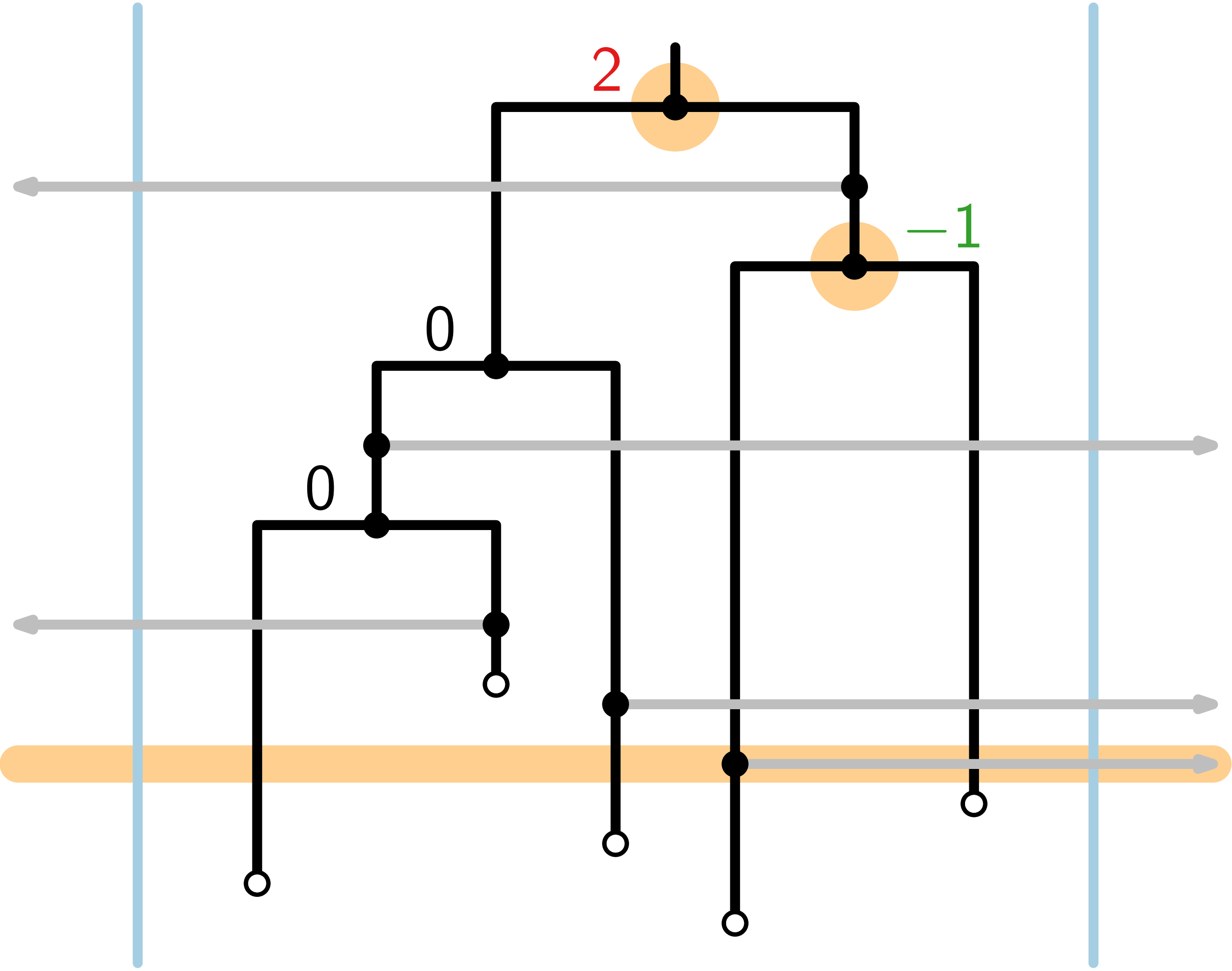
- only rotations of ancestors matter

Subtree Embedding Algorithm



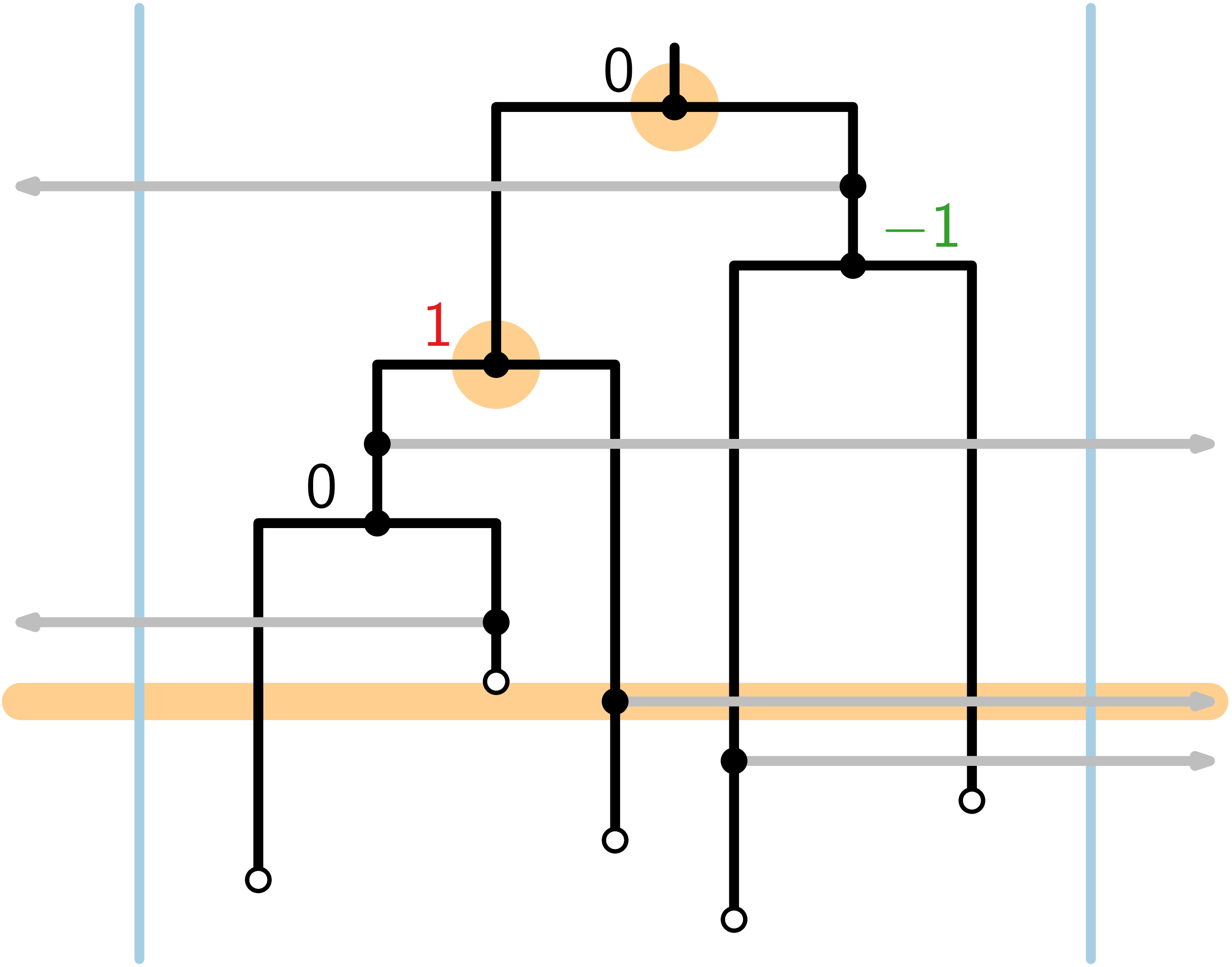
- only rotations of ancestors matter
- store crossing counts

Subtree Embedding Algorithm



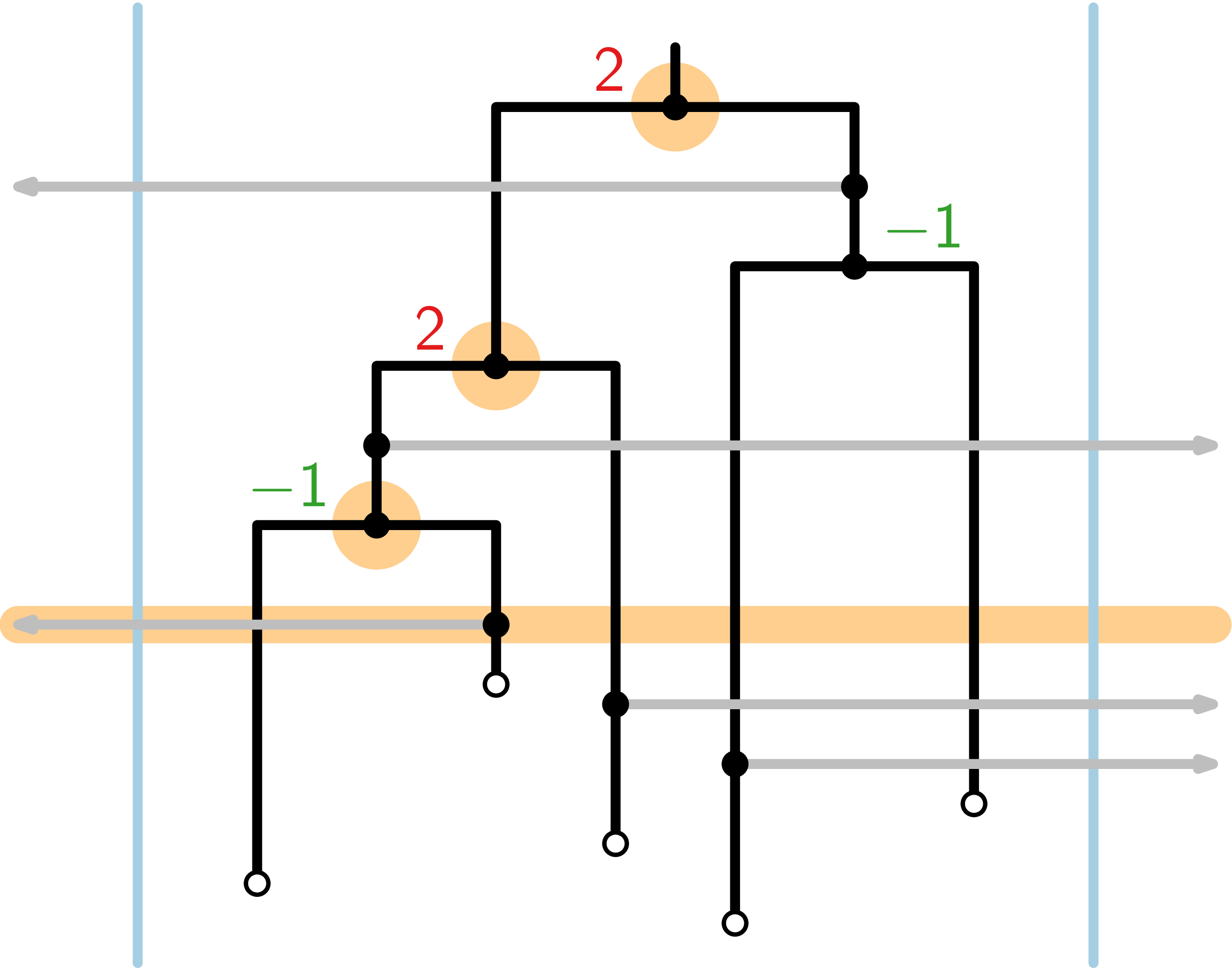
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line

Subtree Embedding Algorithm



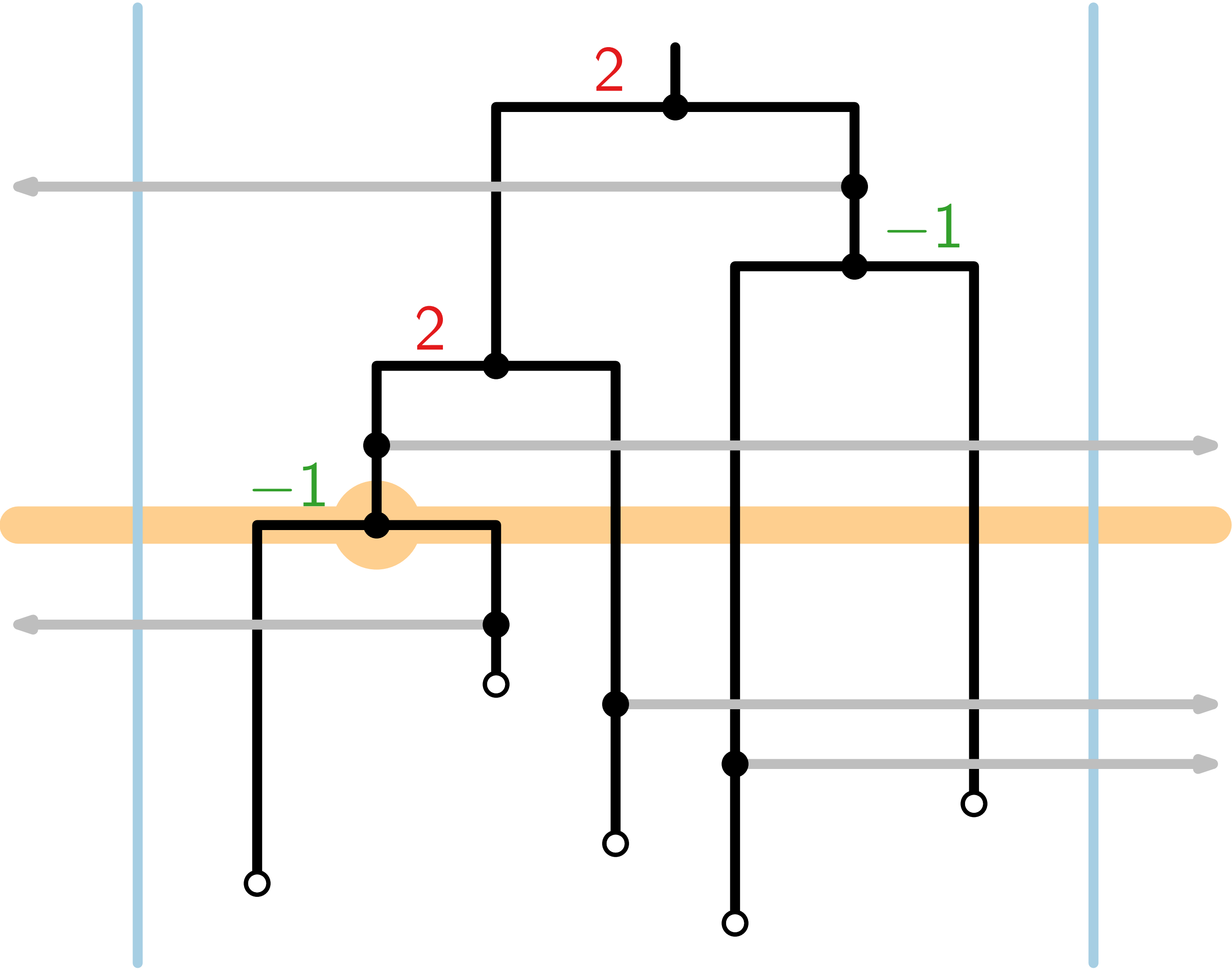
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line

Subtree Embedding Algorithm



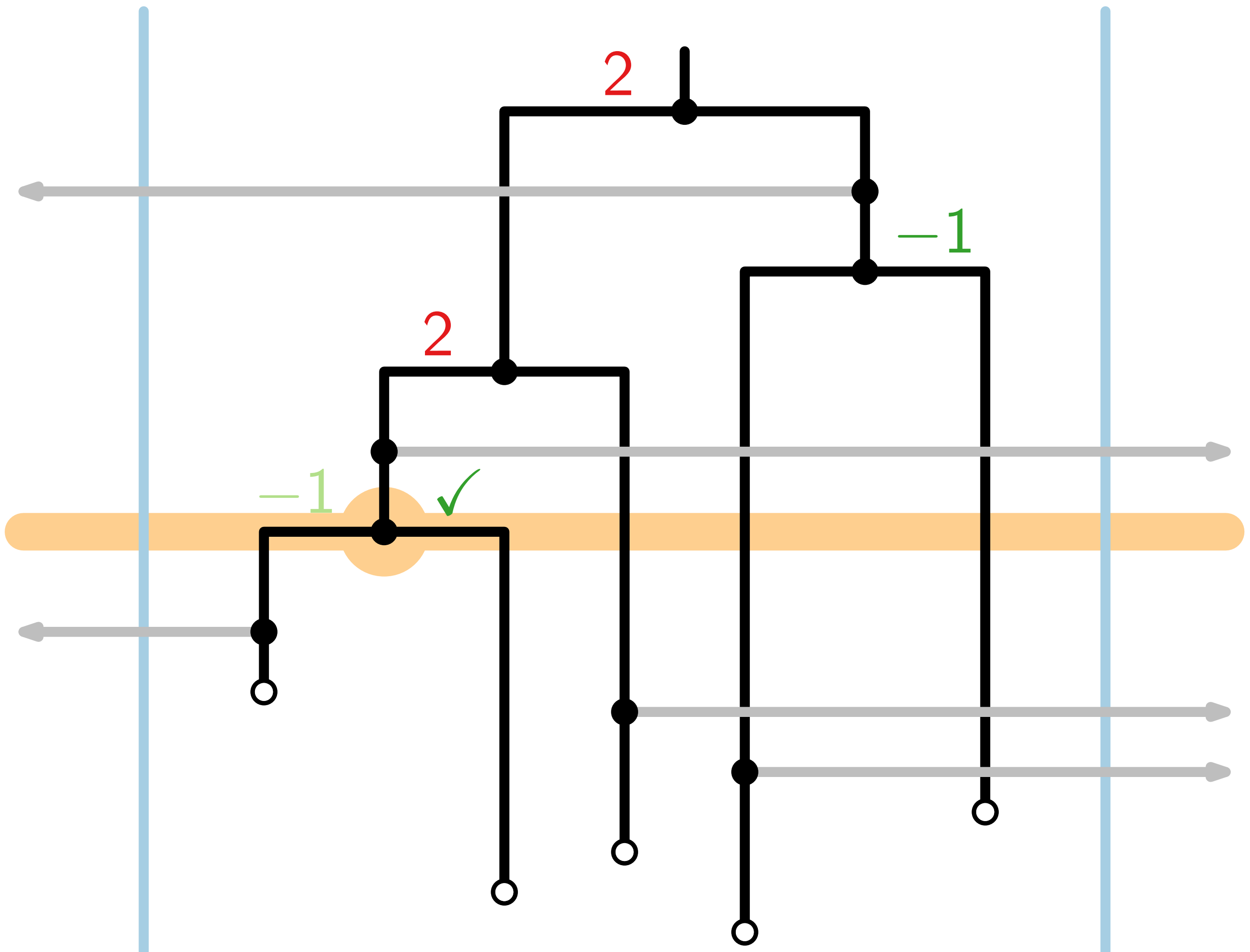
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line

Subtree Embedding Algorithm



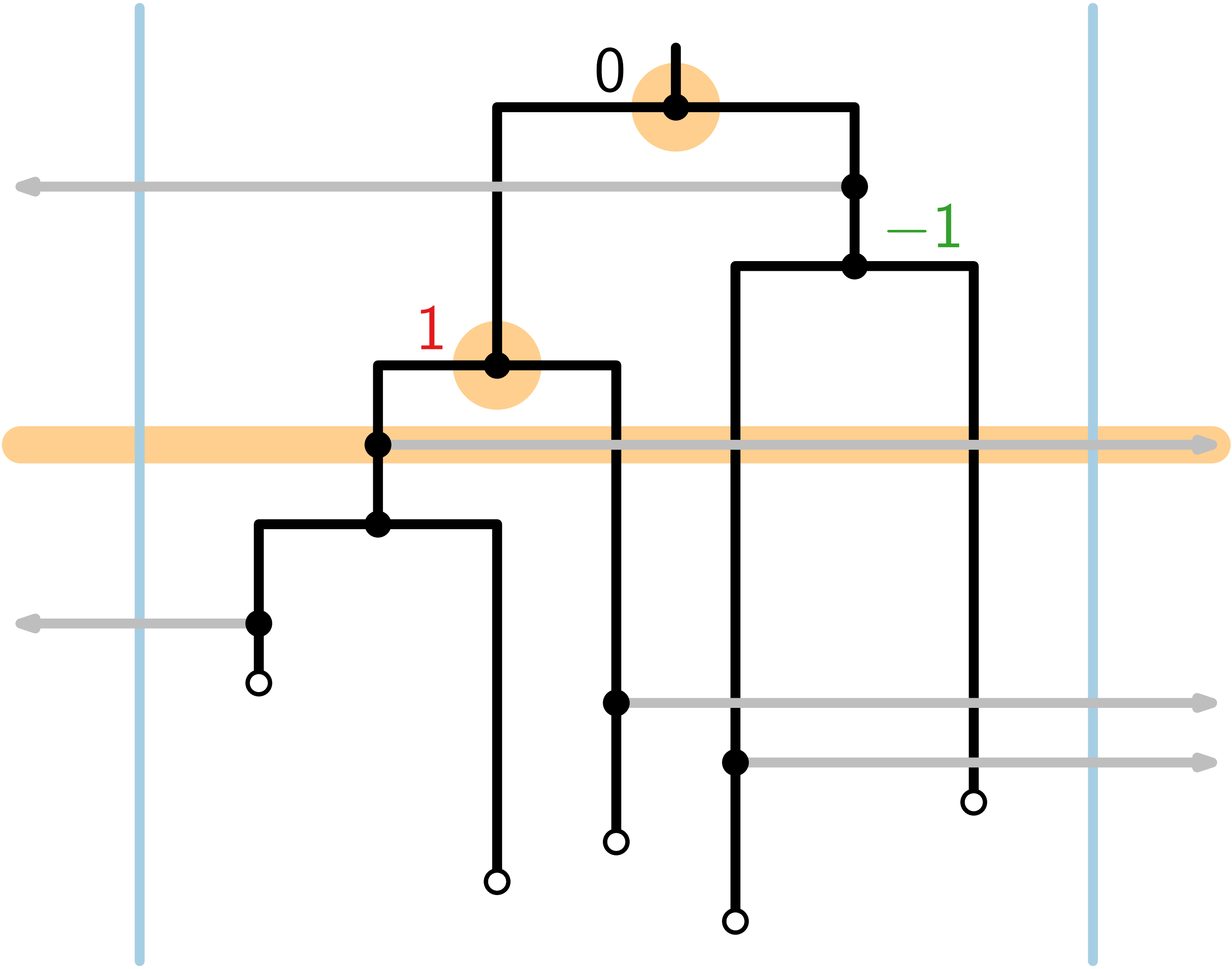
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



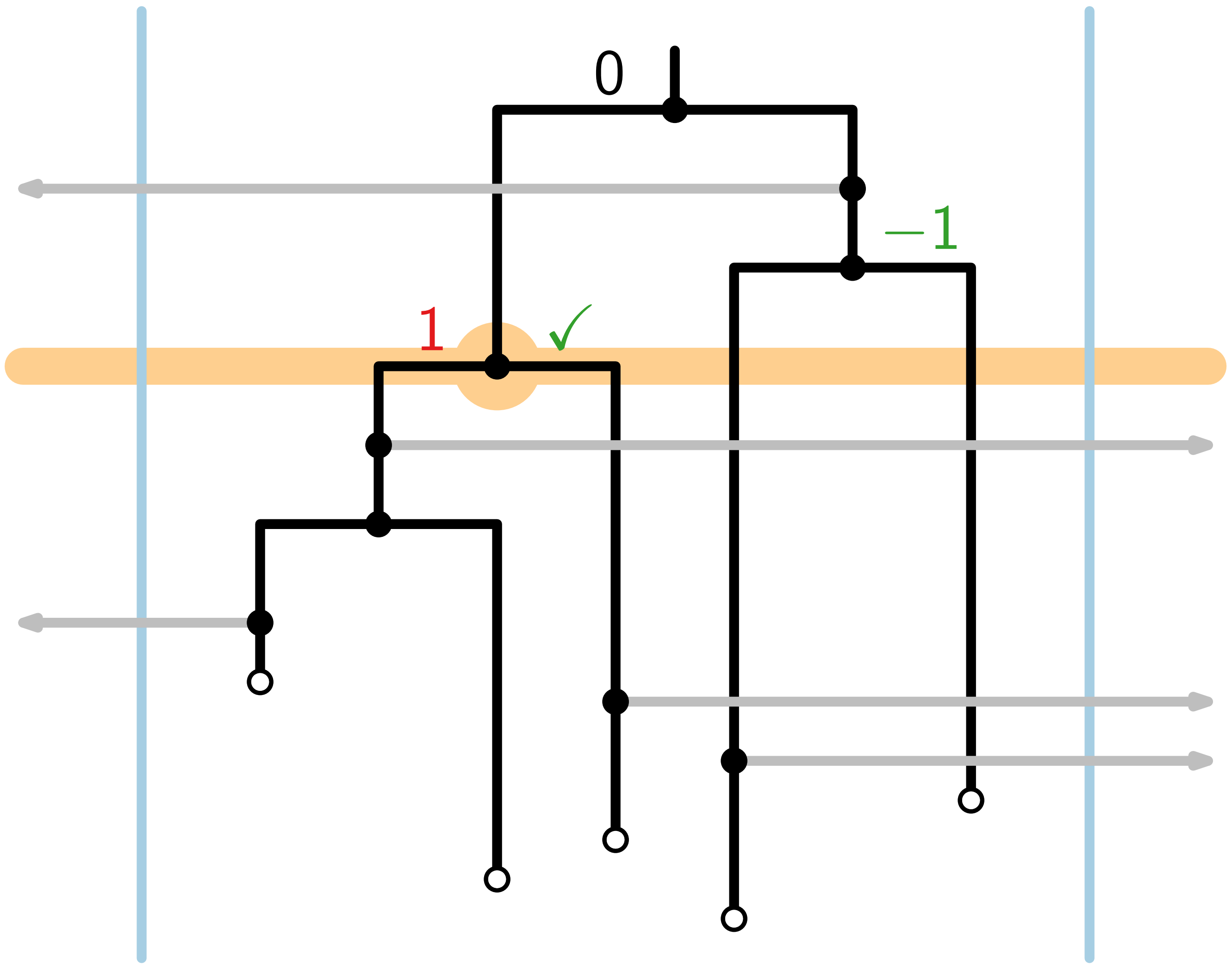
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



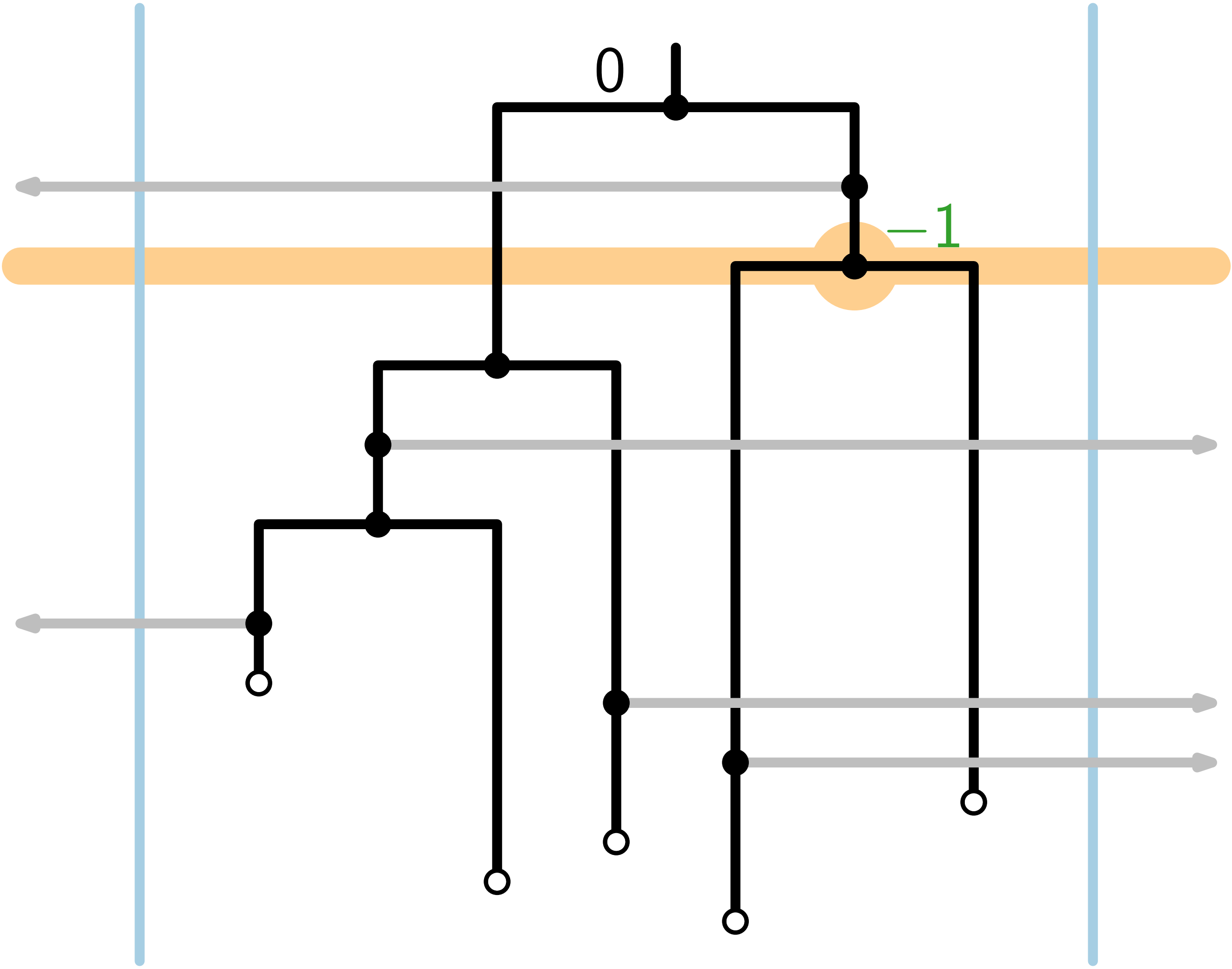
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



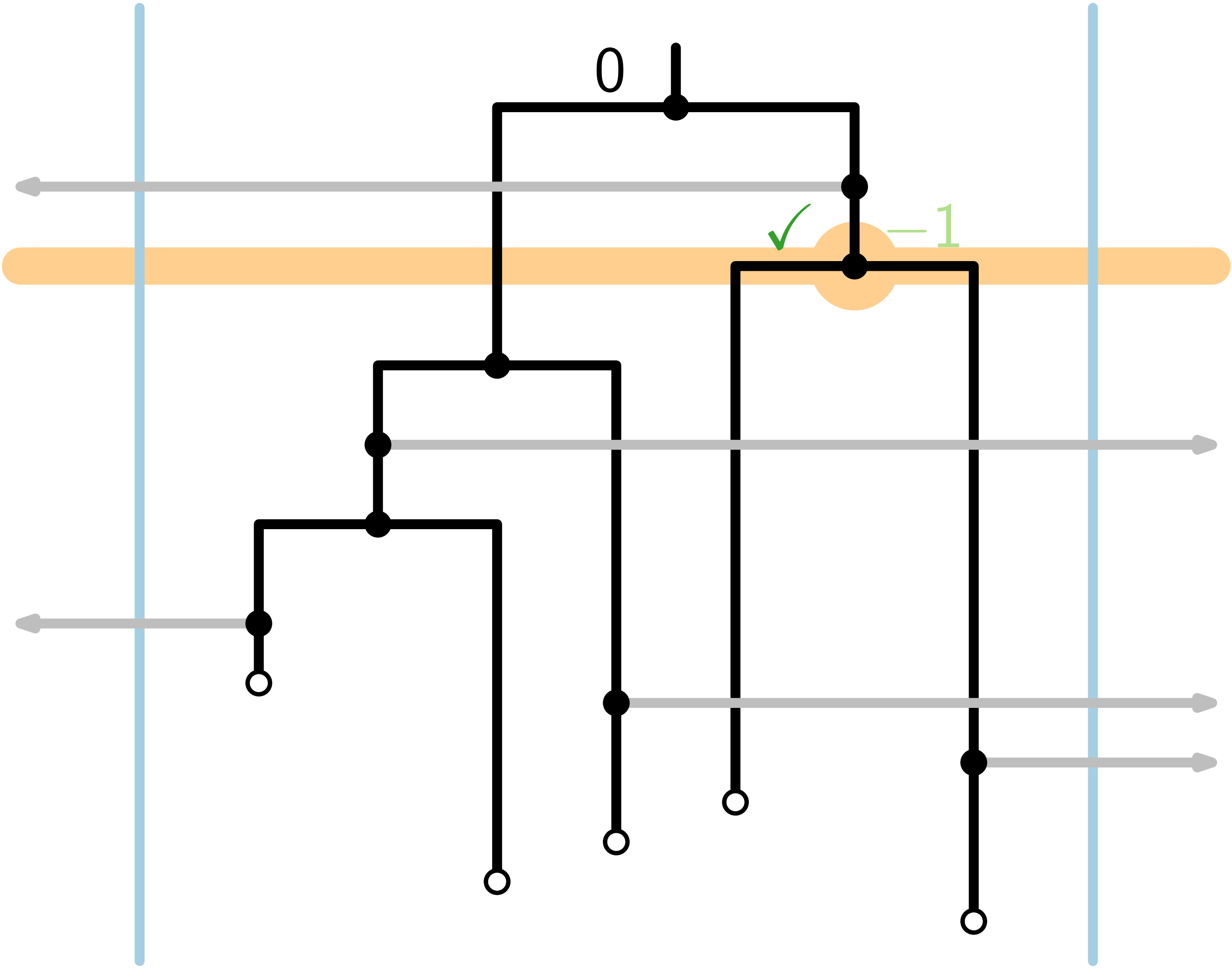
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



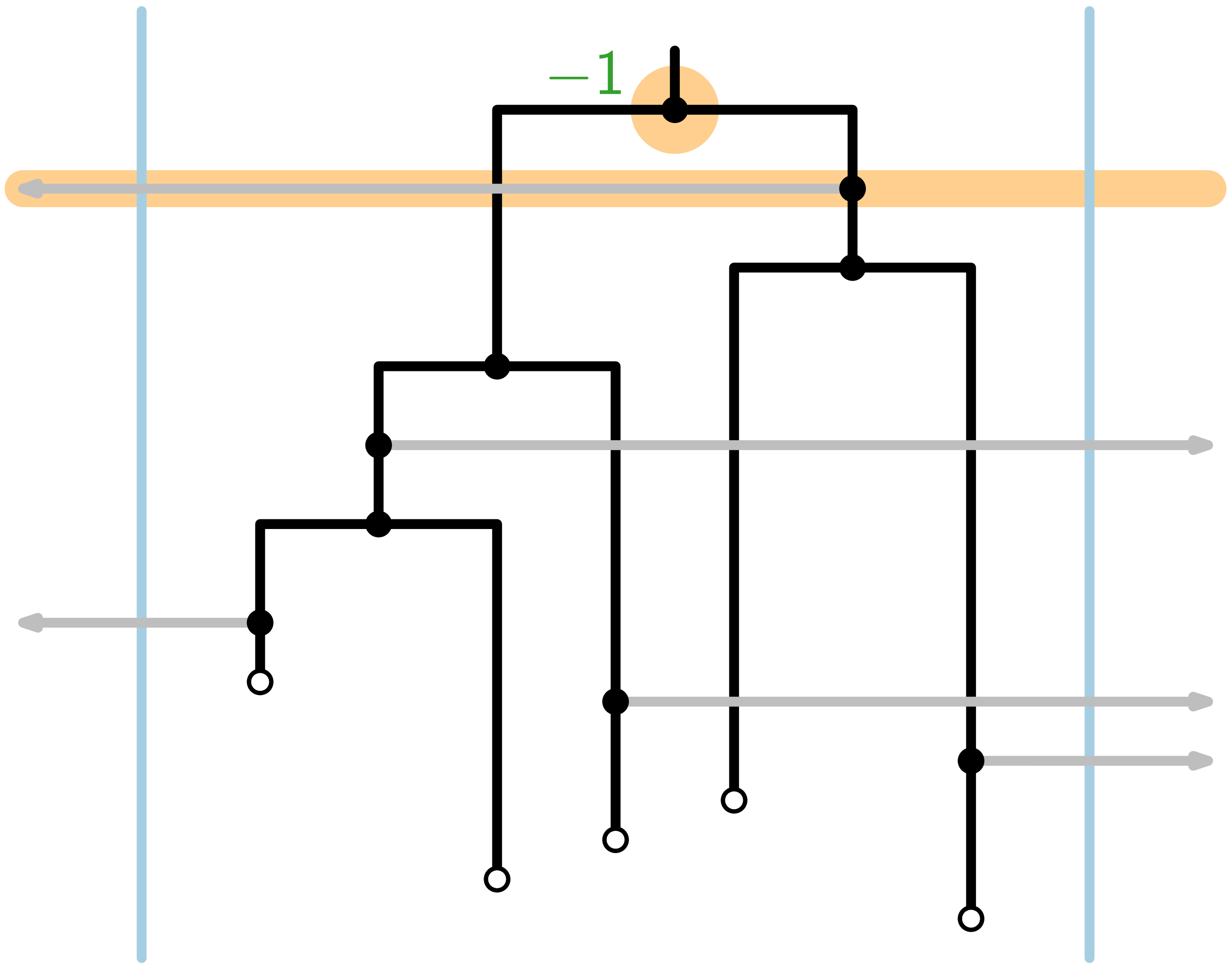
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



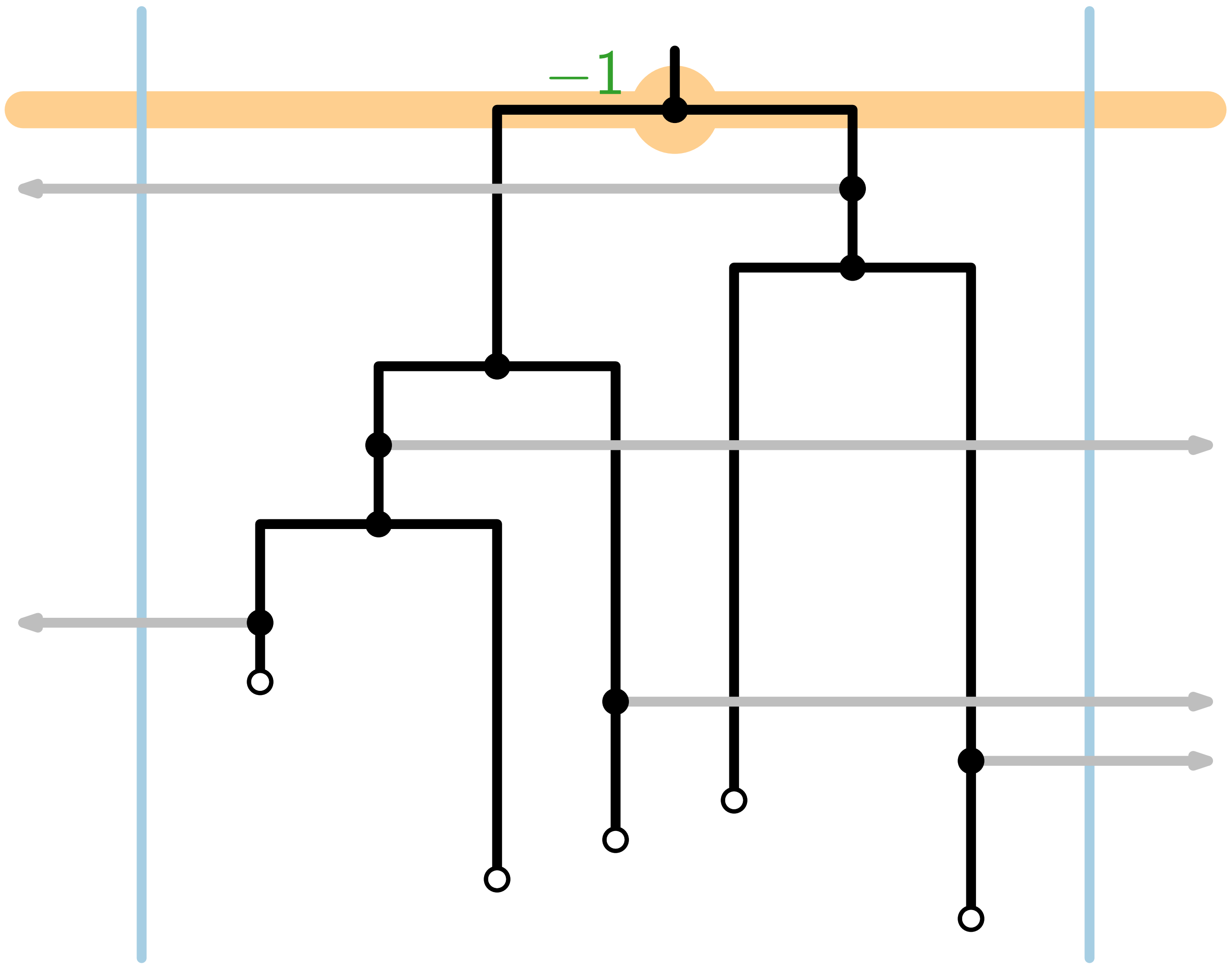
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



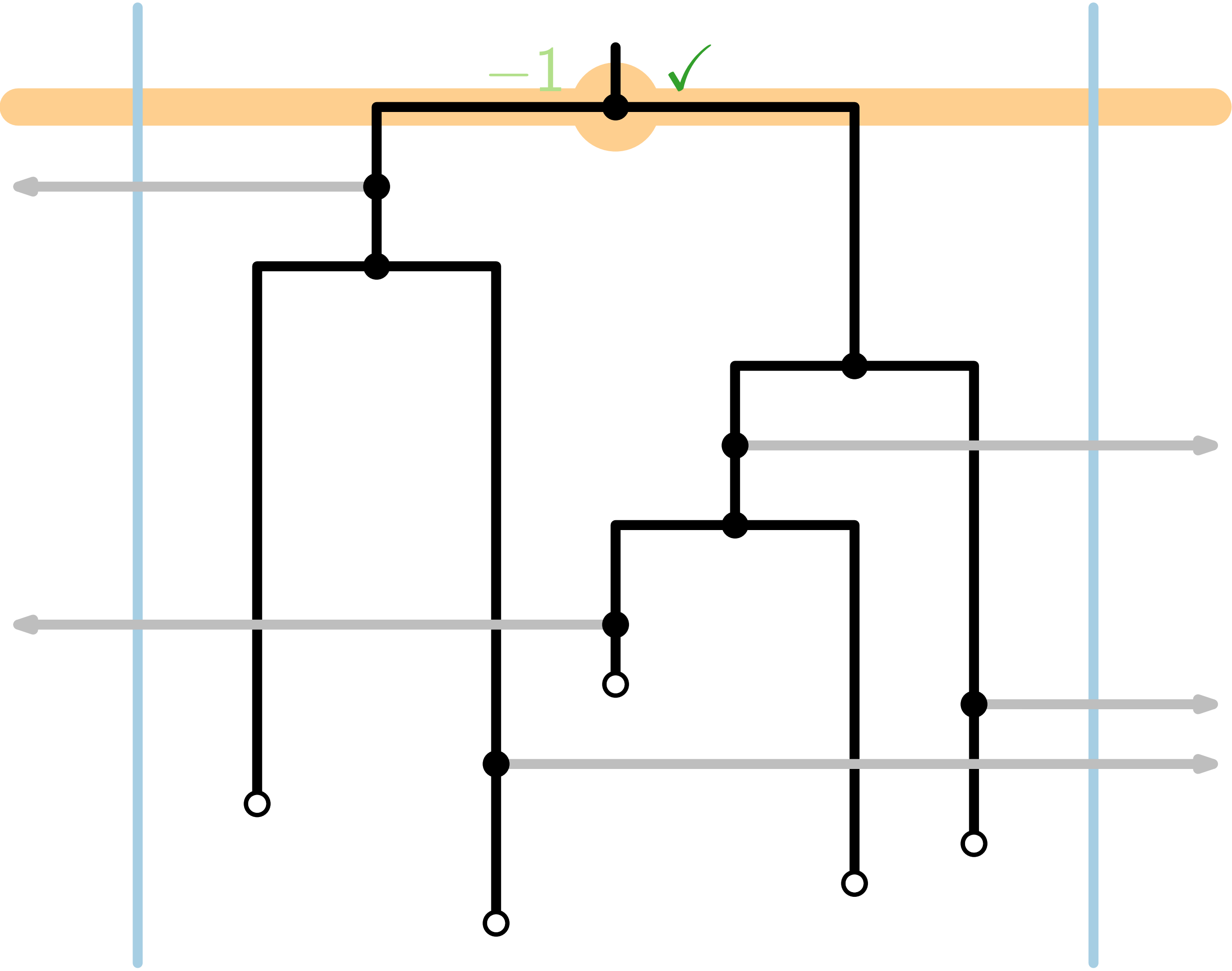
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



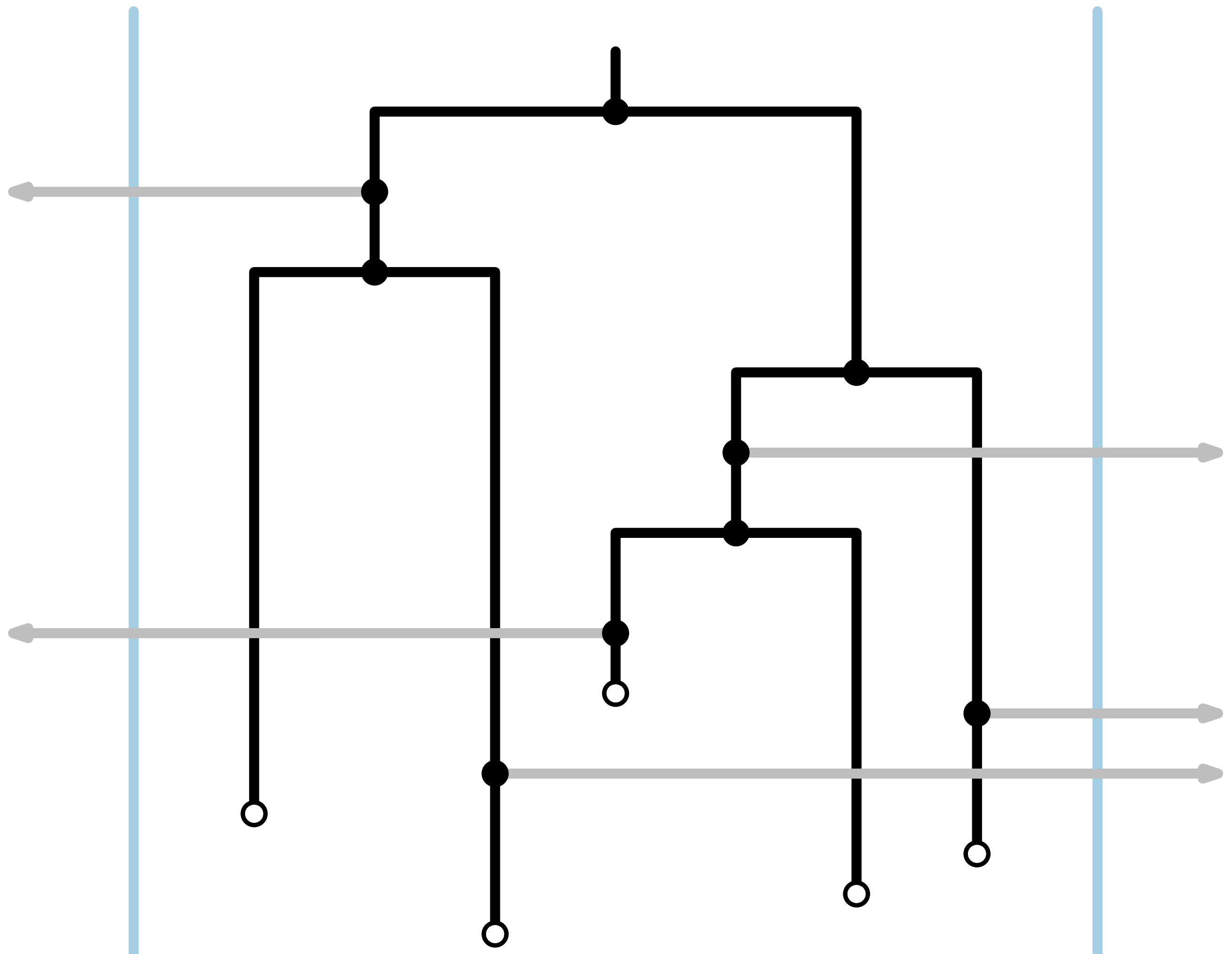
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm



- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

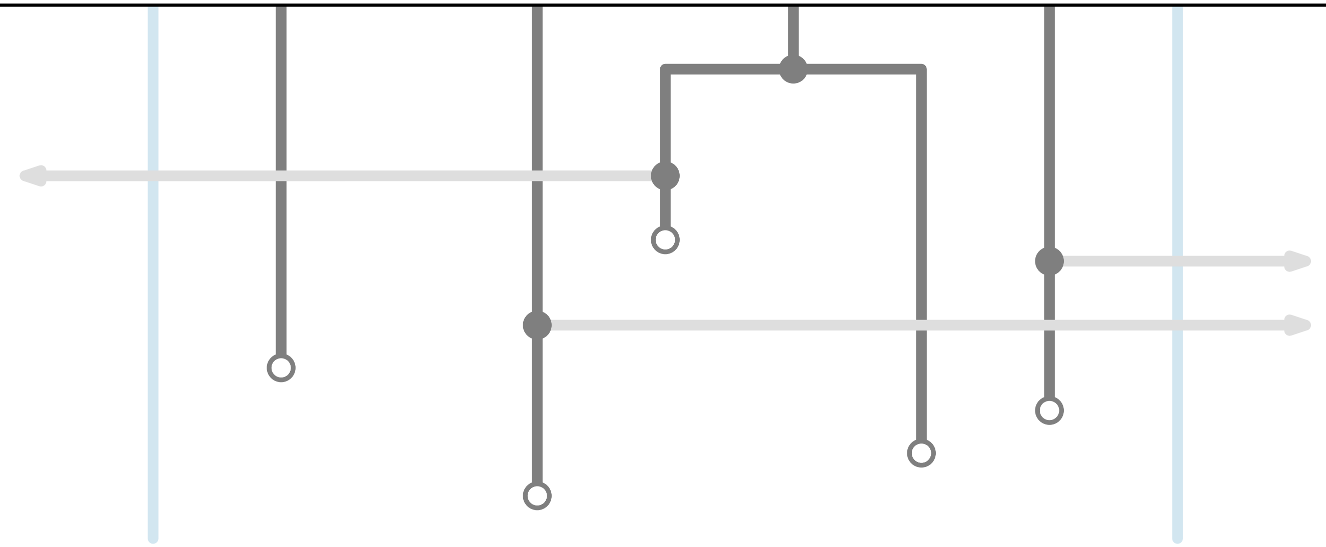
Subtree Embedding Algorithm



- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

Subtree Embedding Algorithm

Lemma.
Can find optimal (binary) subtree embedding in $\mathcal{O}(n^2)$ time;

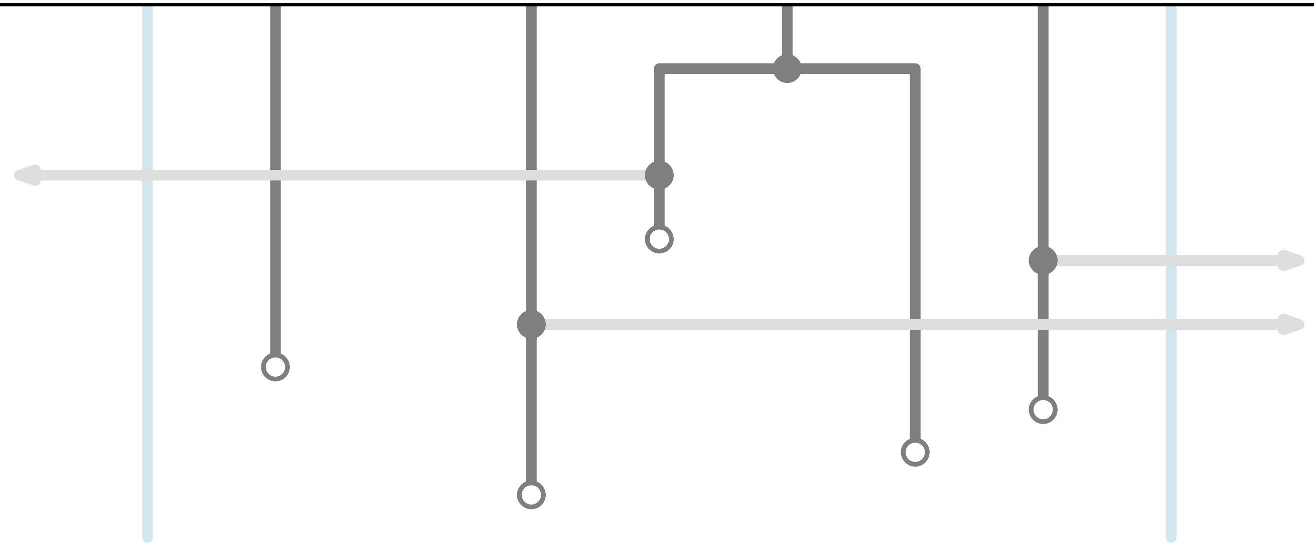


- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

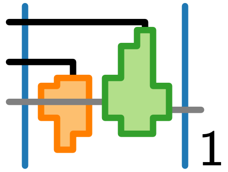
Subtree Embedding Algorithm

Lemma.
Can find optimal (binary) subtree embedding in $\mathcal{O}(n^2)$ time;

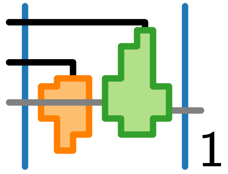
...or if with max degree Δ, then in $\mathcal{O}(\Delta!\Delta n^2)$ time.



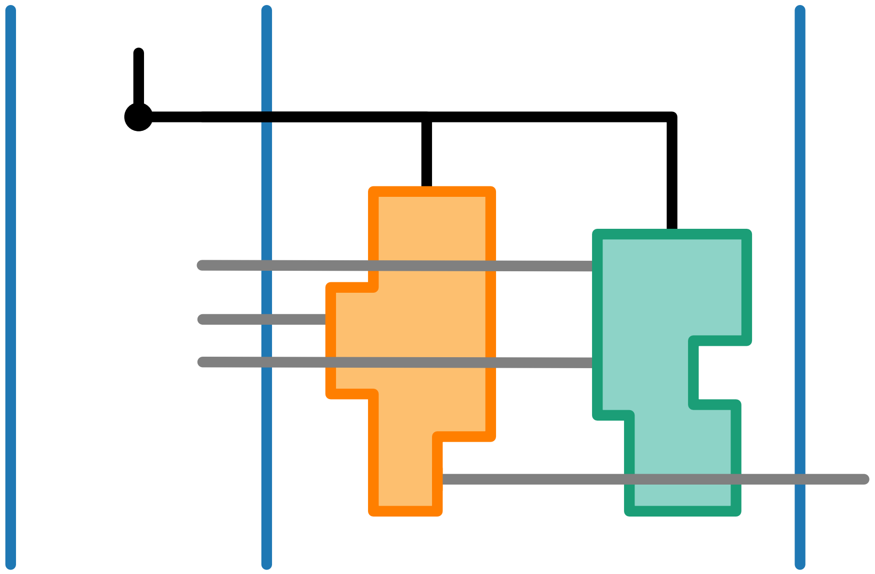
- only rotations of ancestors matter
- store crossing counts
- bottom-up sweep-line
- pick rotation with fewer crossings

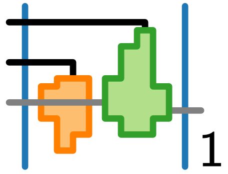
Algorithm for 

- use **subtree embedding** algo for each column subtree

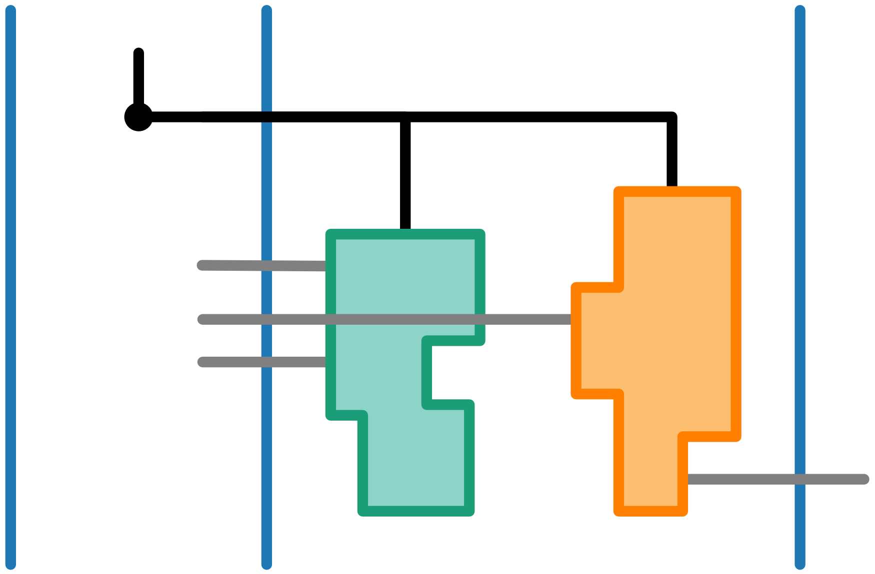
Algorithm for 

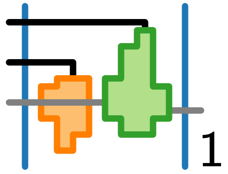
- use **subtree embedding** algo for each column subtree
- for **subtree arrangement**,



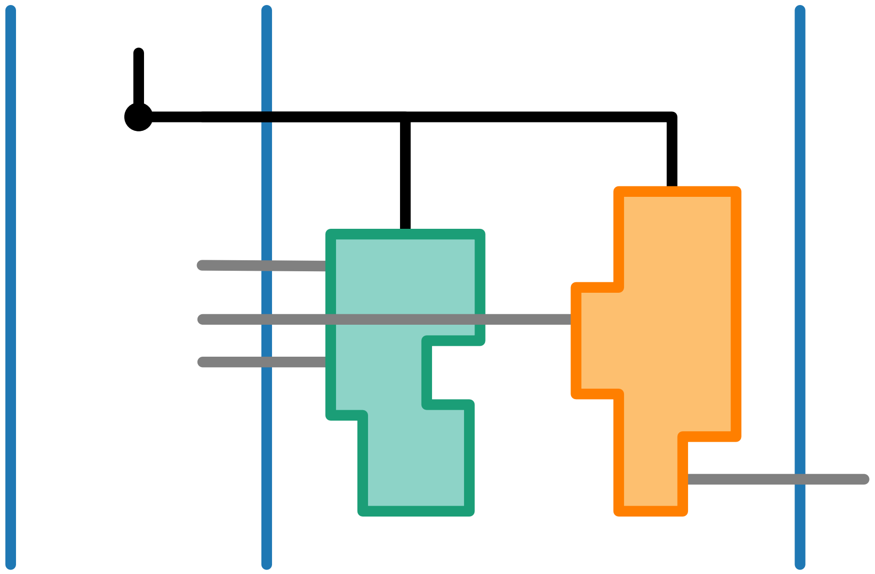
Algorithm for 

- use **subtree embedding** algo for each column subtree
- for **subtree arrangement**, try both (all) orders of column subtrees with same parent

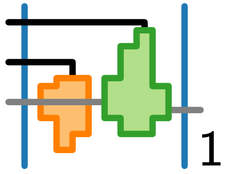
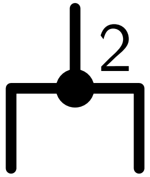


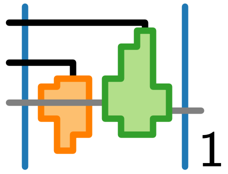
Algorithm for 

- use **subtree embedding** algo for each column subtree
- for **subtree arrangement**, try both (all) orders of column subtrees with same parent

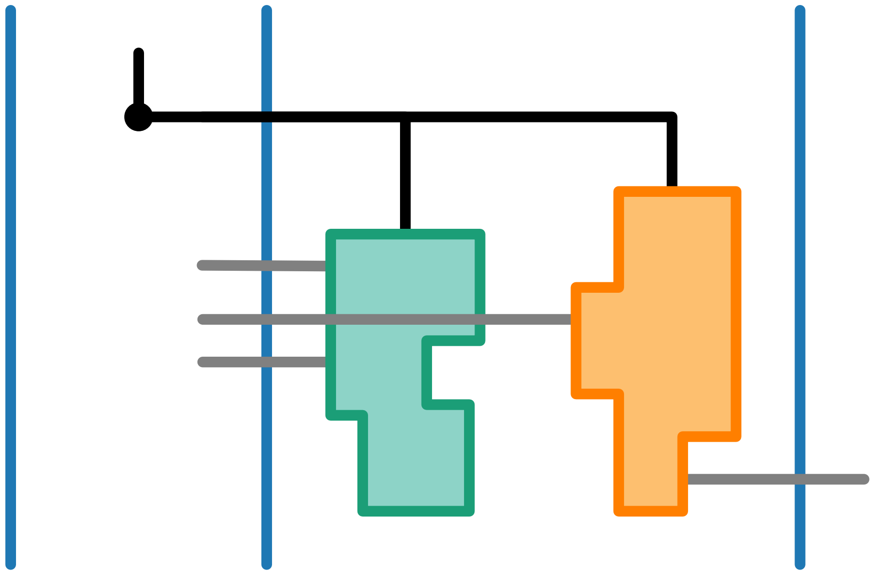


Theorem.

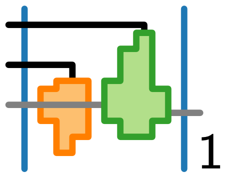
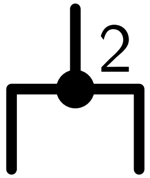
 +  → $\mathcal{O}(n^2)$-time algorithm

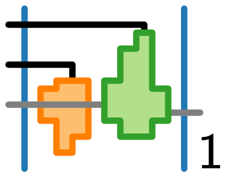
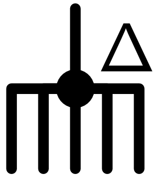
Algorithm for 

- use **subtree embedding** algo for each column subtree
- for **subtree arrangement**, try both (all) orders of column subtrees with same parent



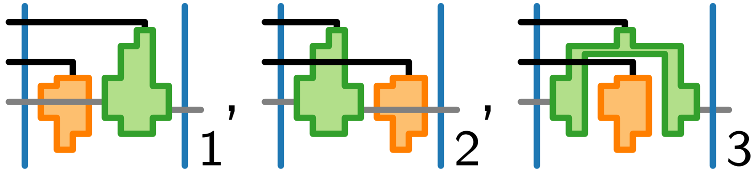
Theorem.

 $+$  → $\mathcal{O}(n^2)$-time algorithm

 $+$  → $\mathcal{O}(\Delta!\Delta n^2)$-time algorithm

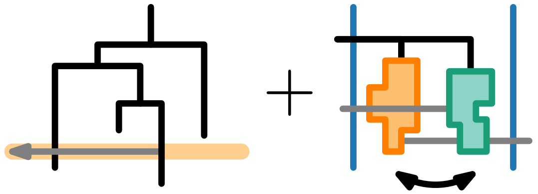
Overview

Drawing Style



Crossing Minimisation

P

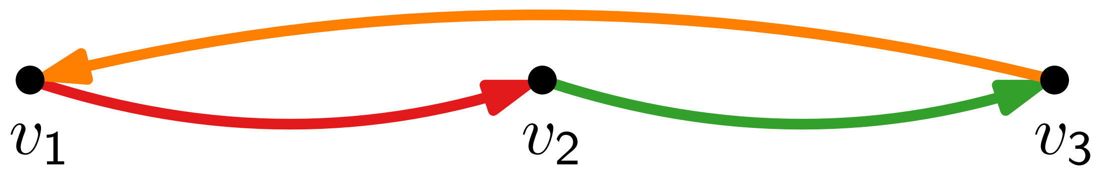


NP

FPT

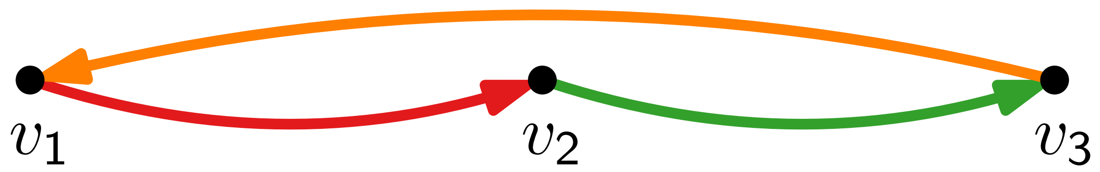
NP-hardness

Feedback Arc Set
instance



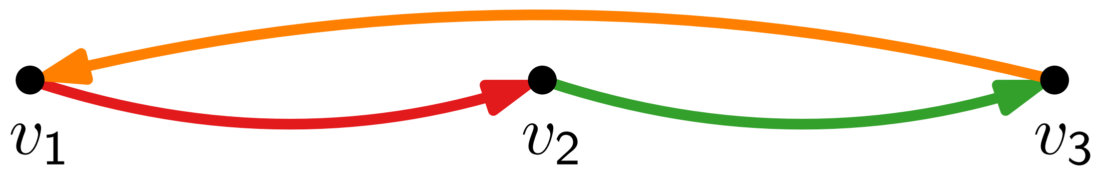
NP-hardness

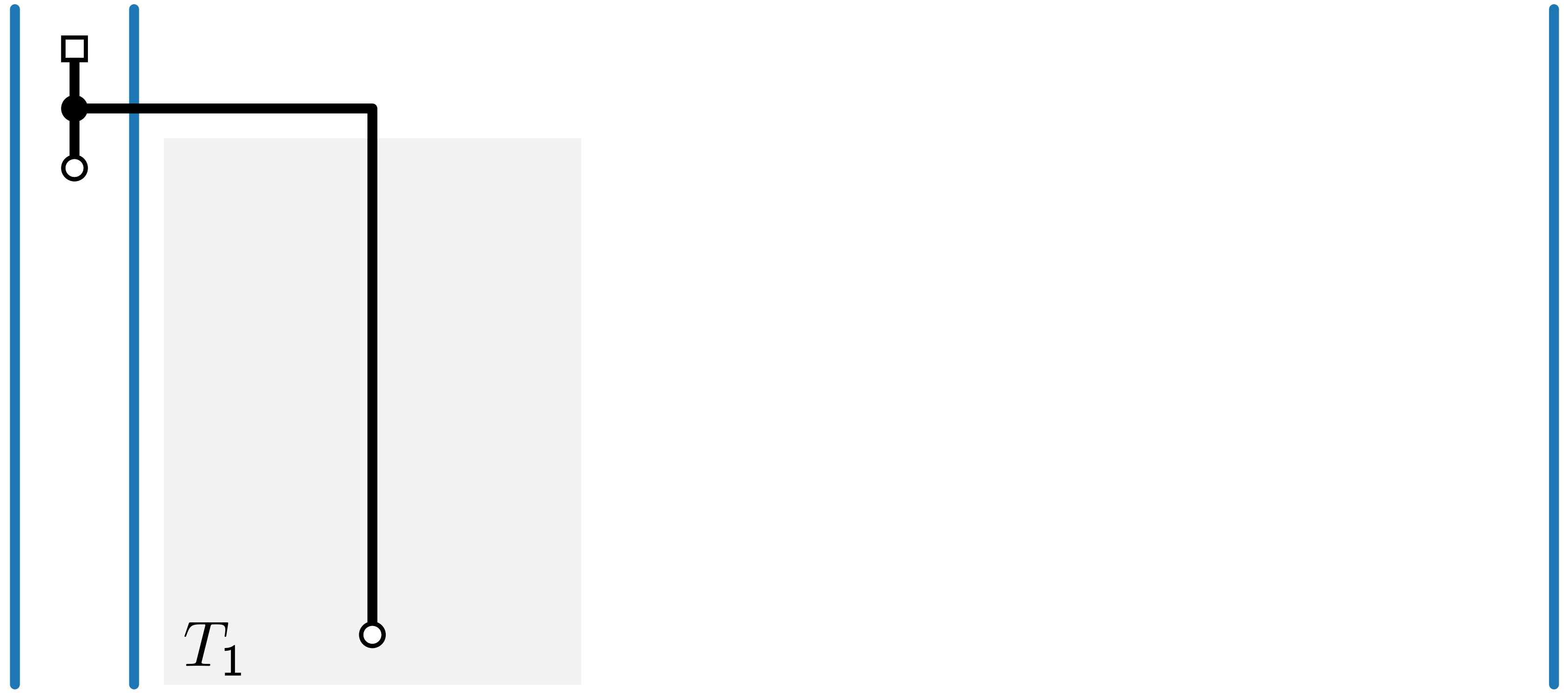
Feedback Arc Set
instance



NP-hardness

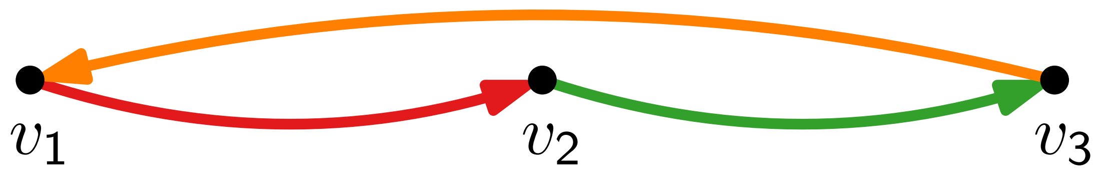
Feedback Arc Set instance

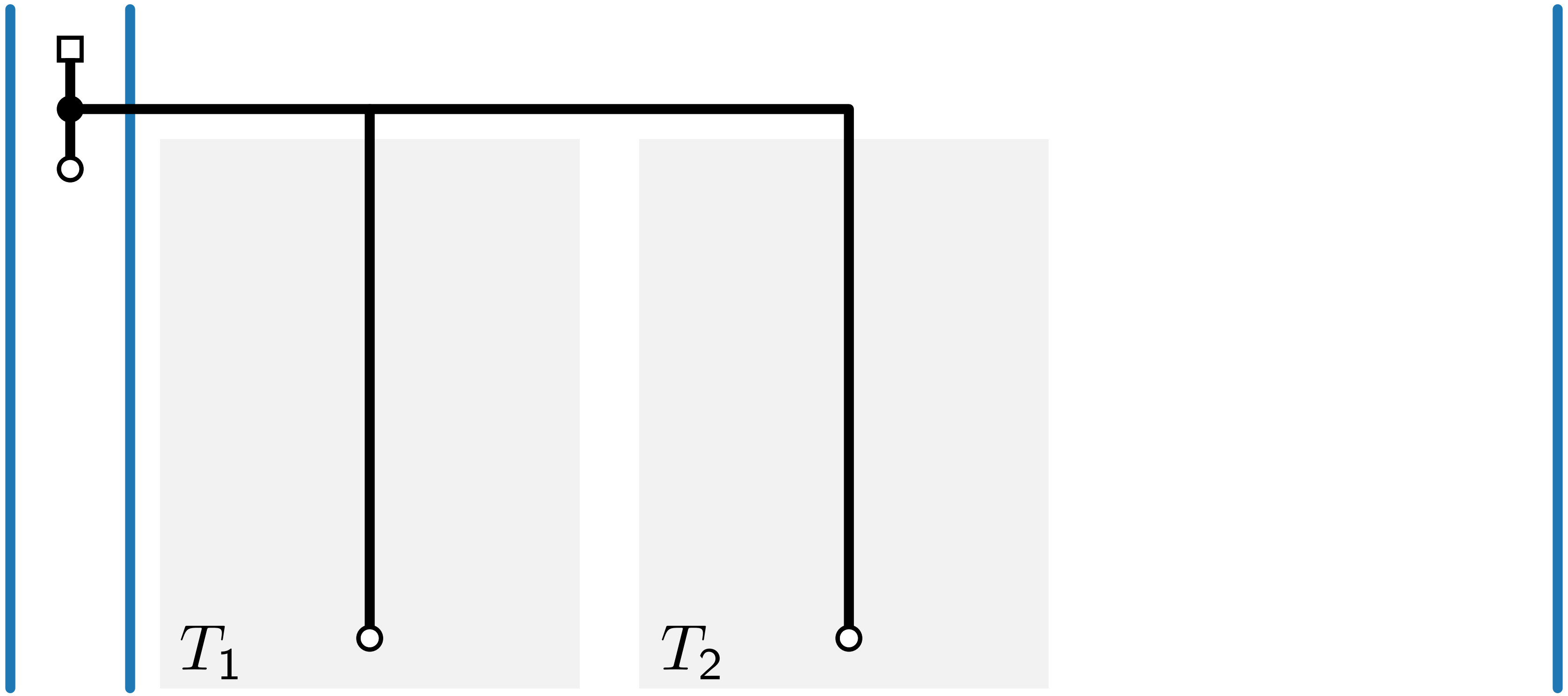




NP-hardness

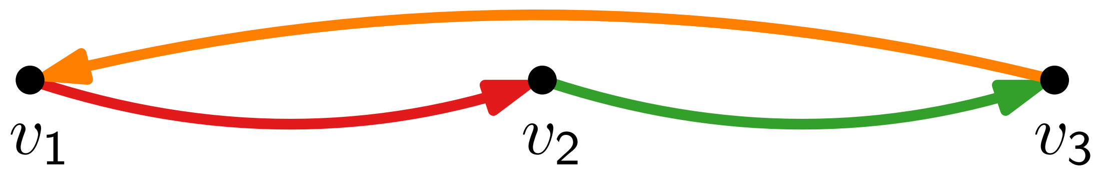
Feedback Arc Set
instance

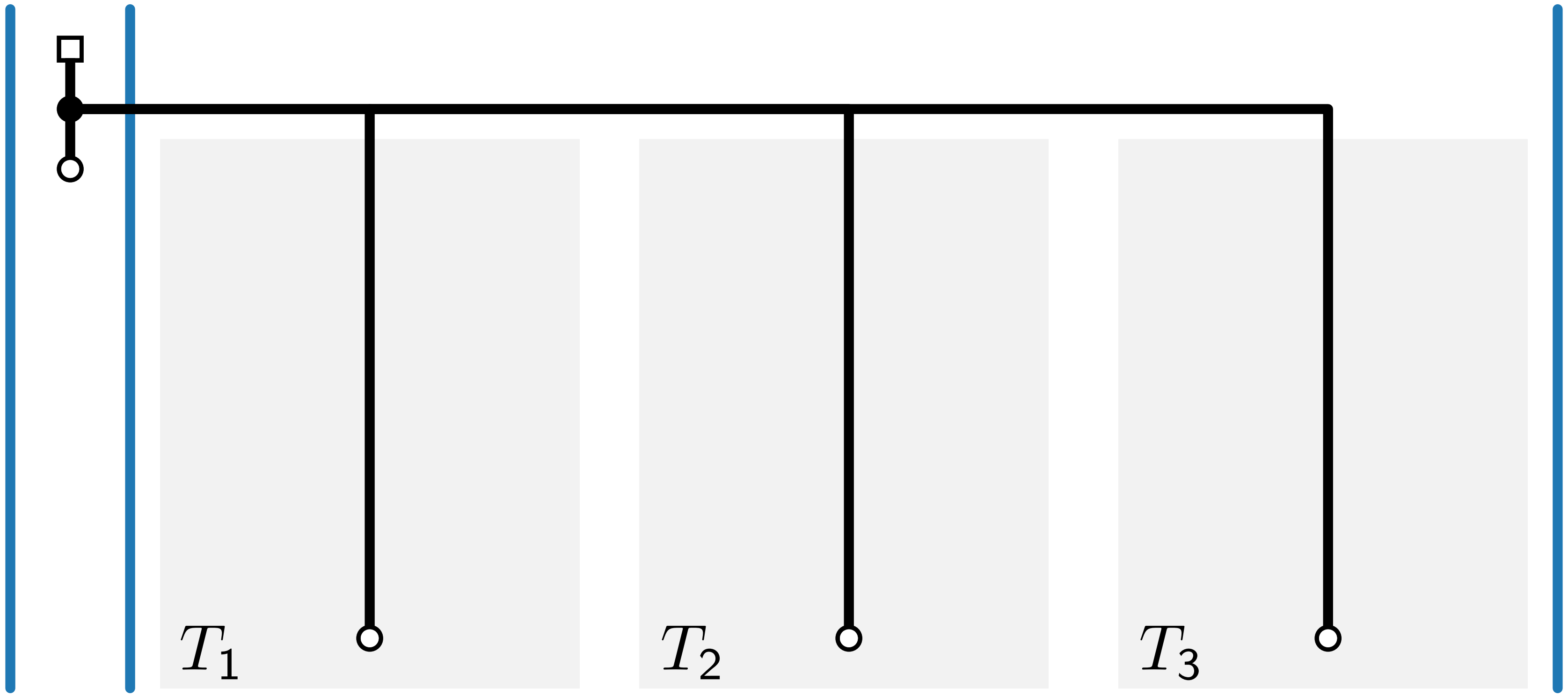




NP-hardness

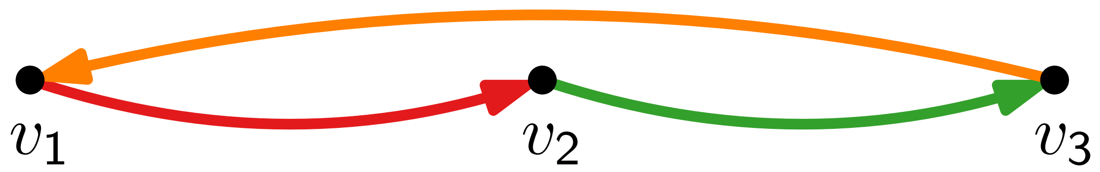
Feedback Arc Set
instance

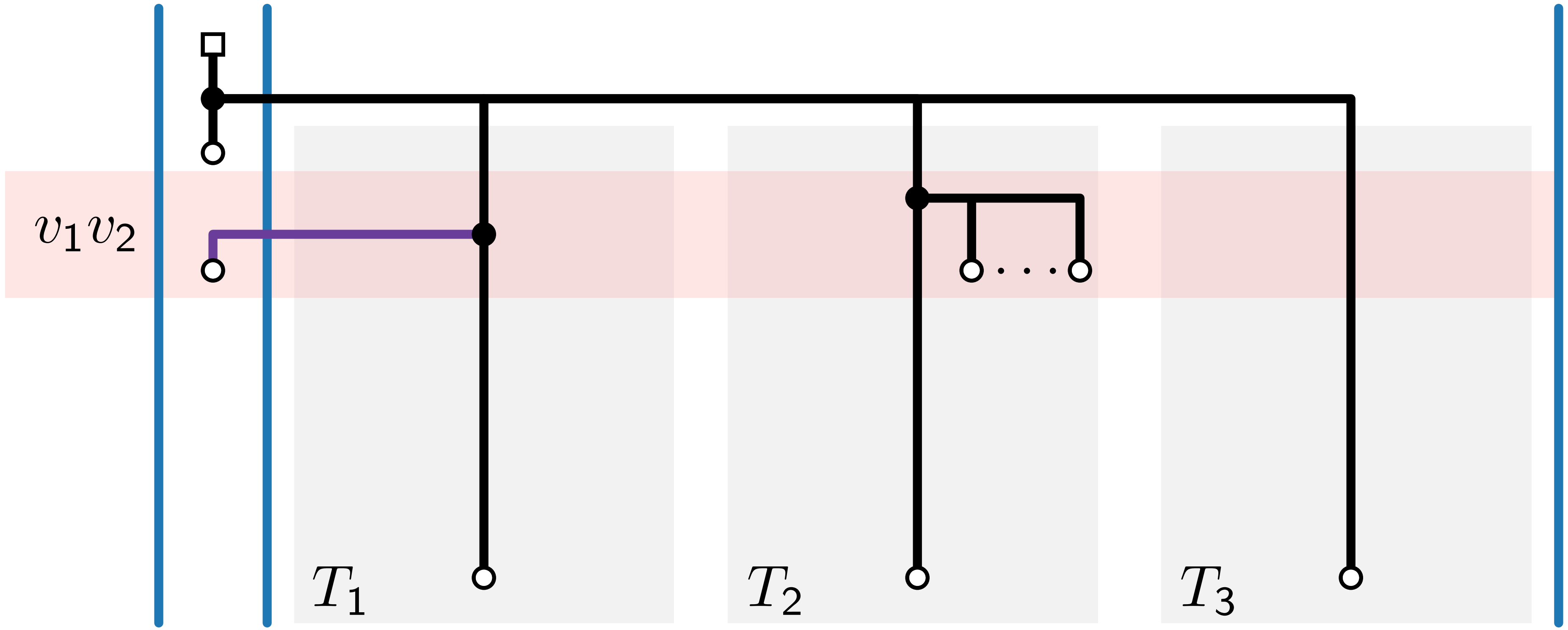




NP-hardness

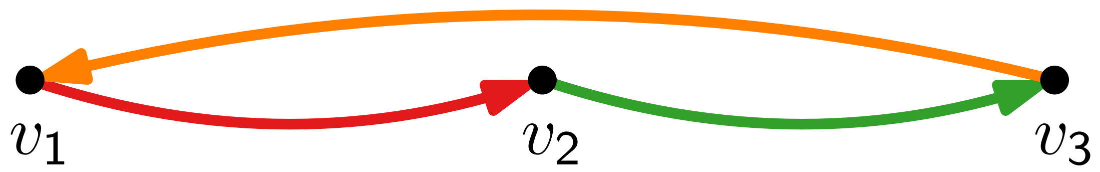
Feedback Arc Set instance

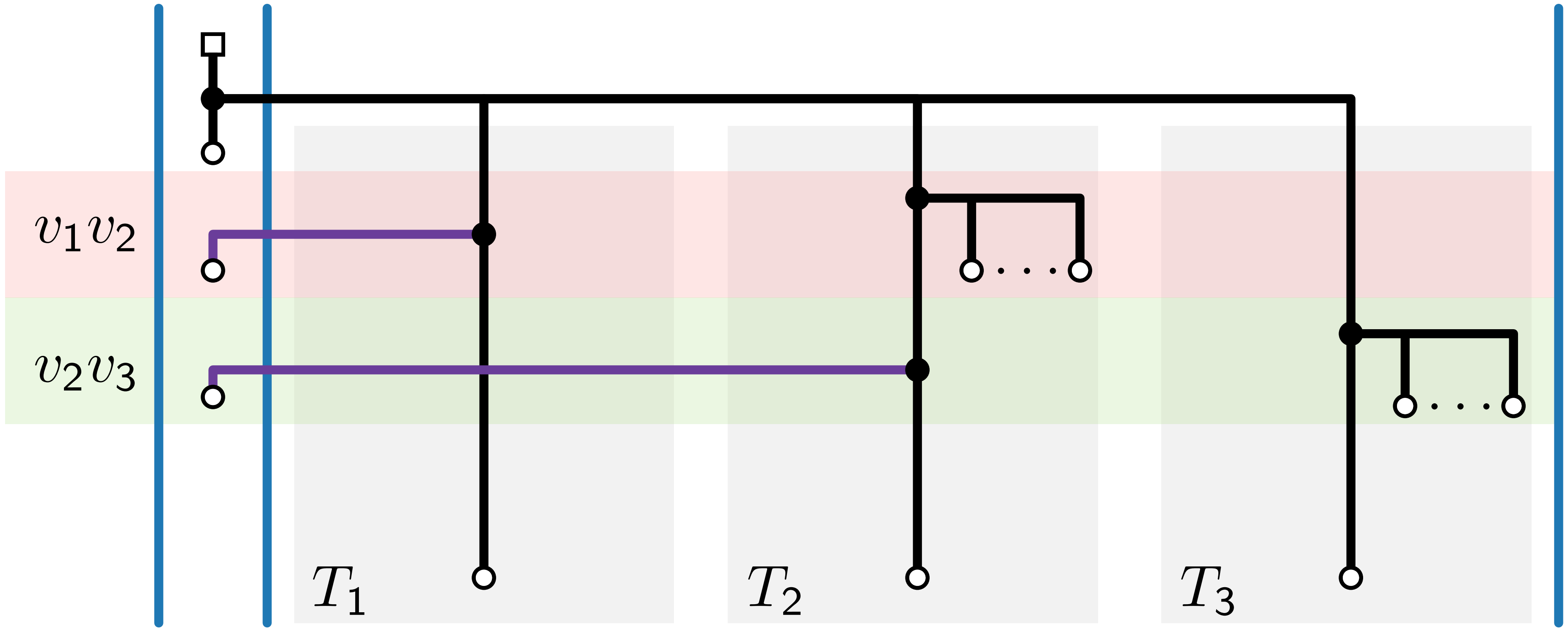




NP-hardness

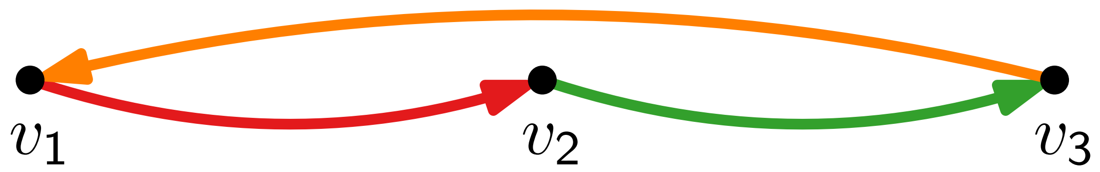
Feedback Arc Set instance

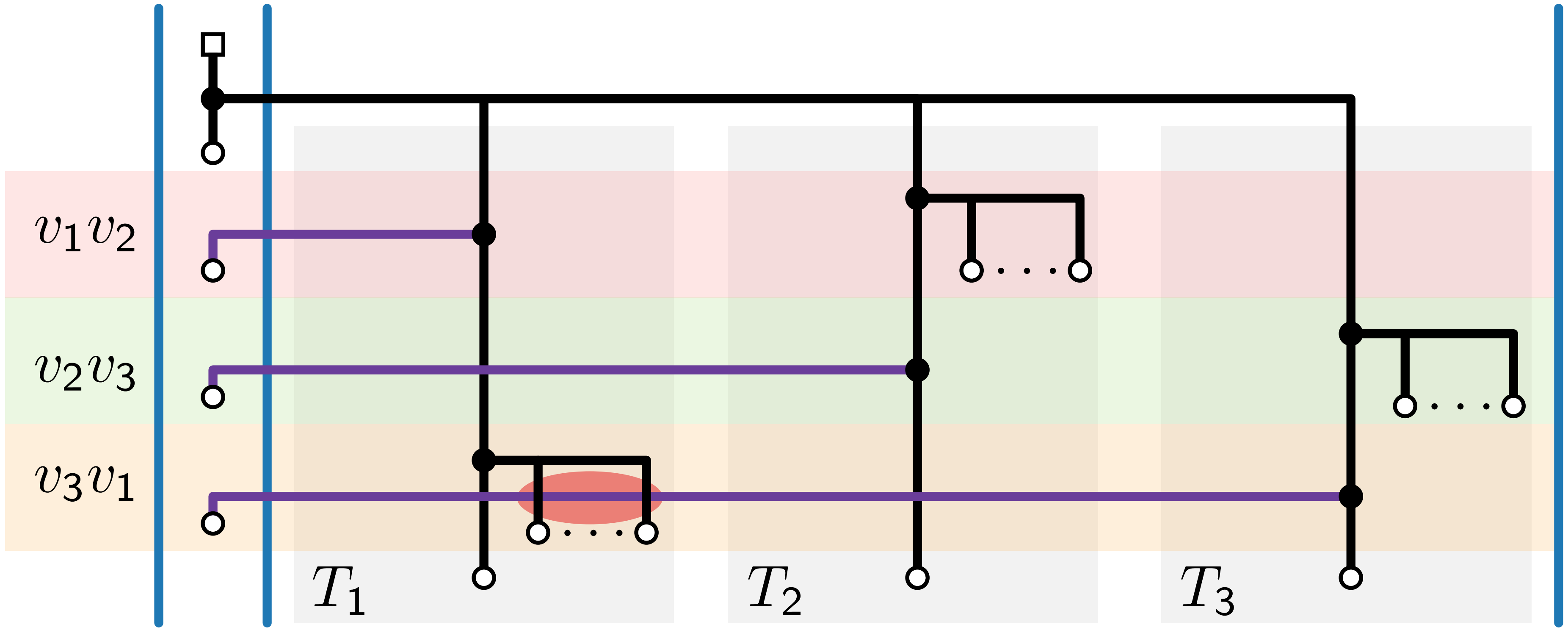




NP-hardness

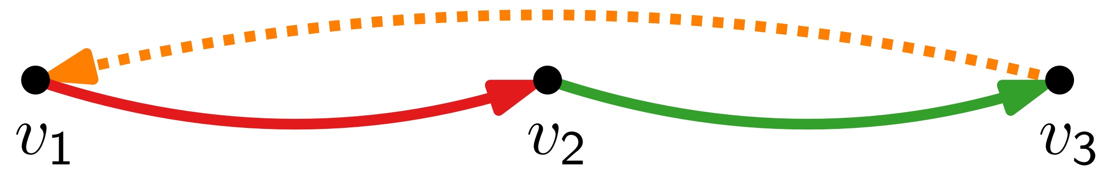
Feedback Arc Set
instance

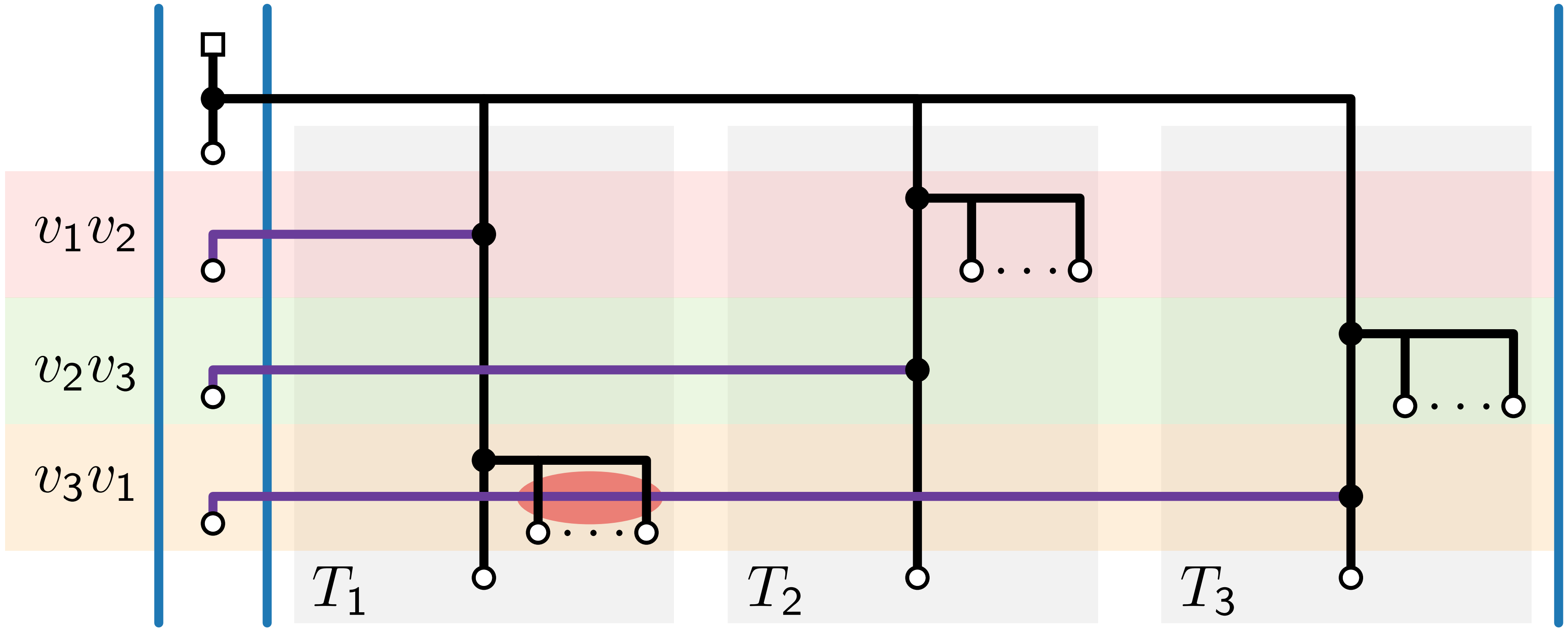




NP-hardness

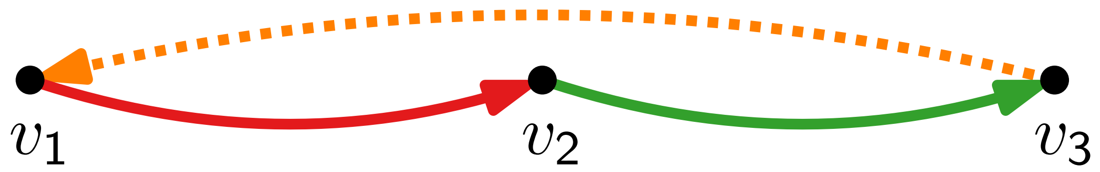
Feedback Arc Set instance

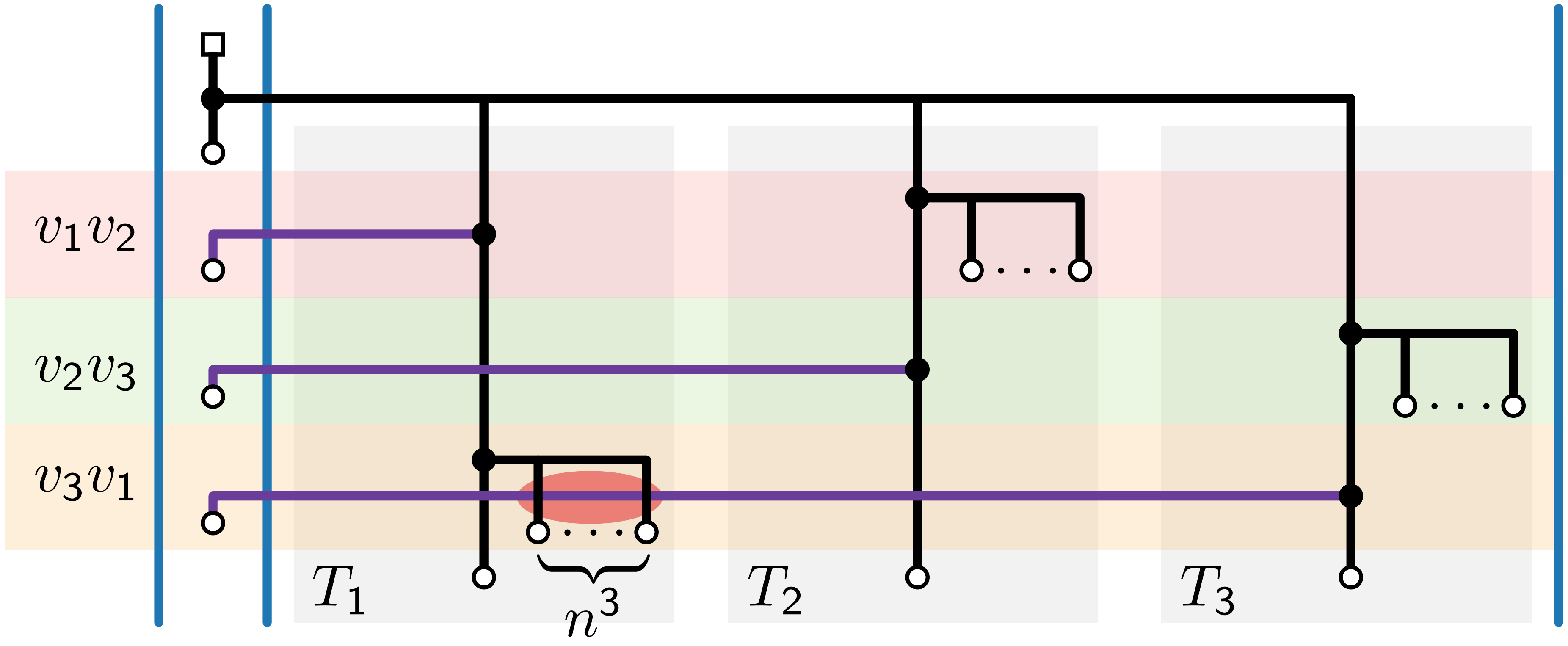




NP-hardness

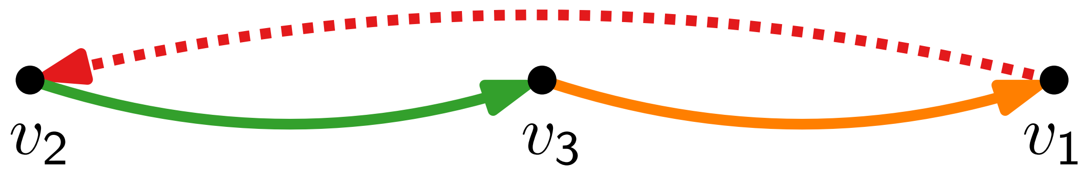
Feedback Arc Set
instance

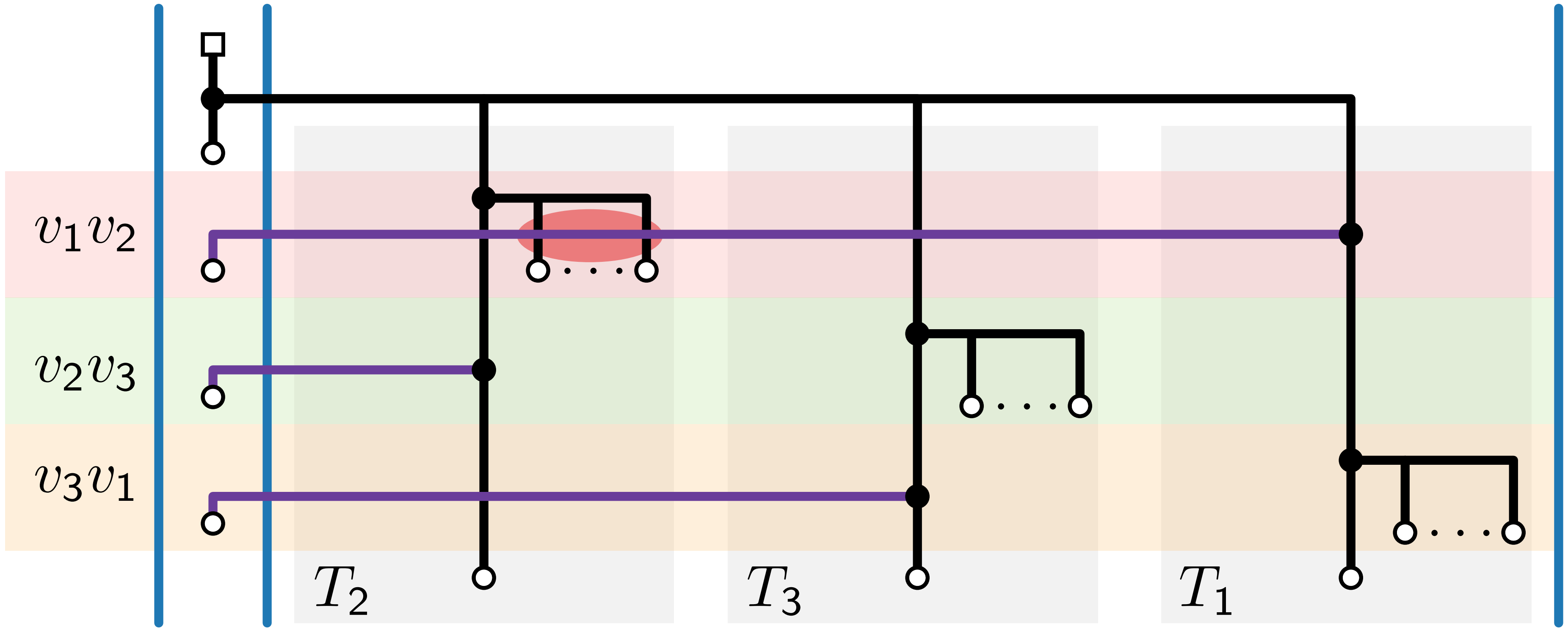




NP-hardness

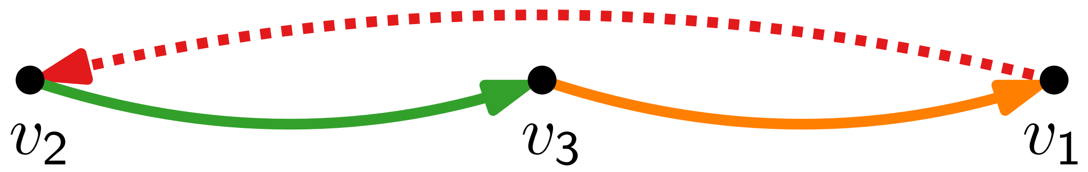
Feedback Arc Set
instance

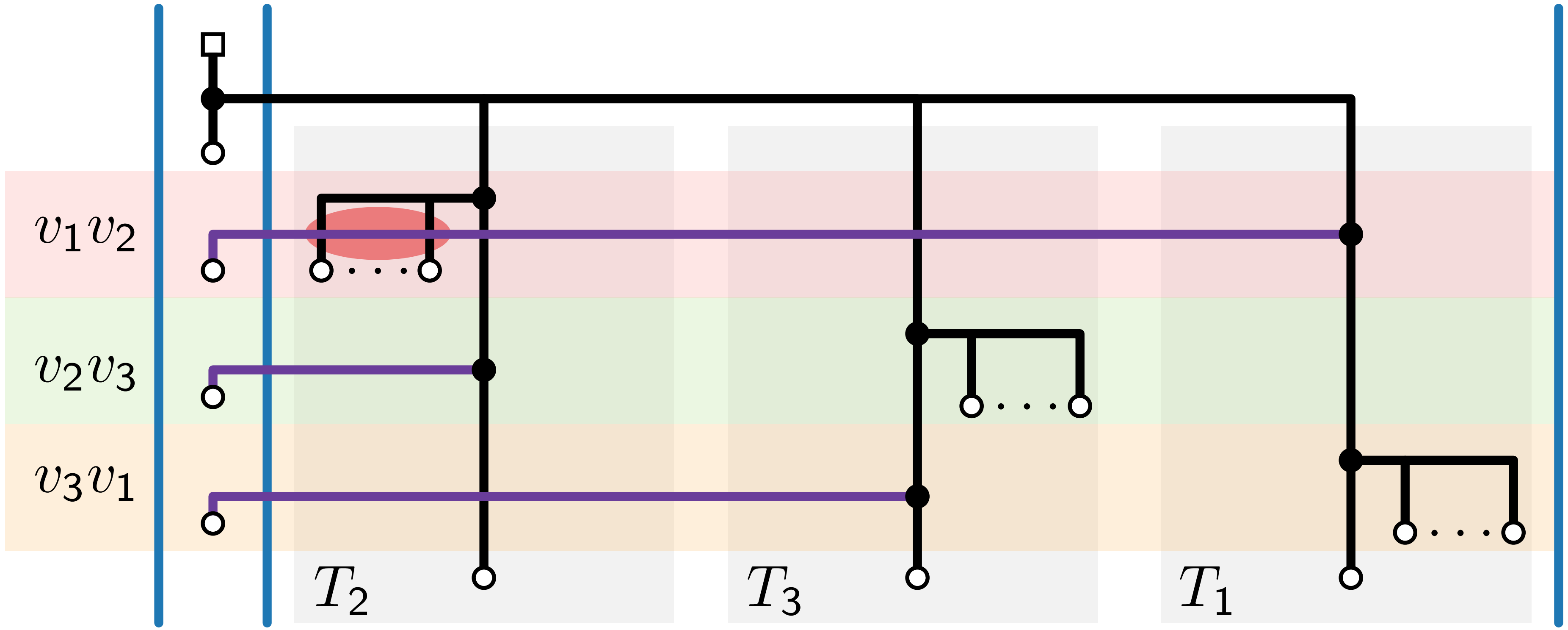




NP-hardness

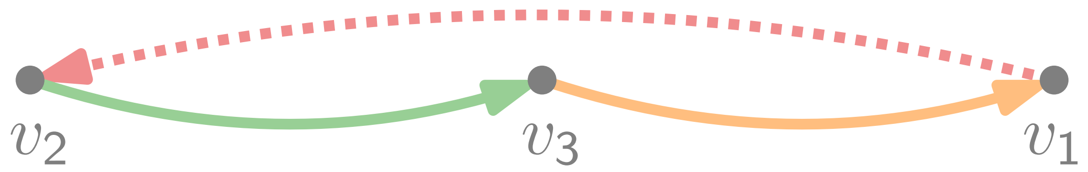
Feedback Arc Set
instance

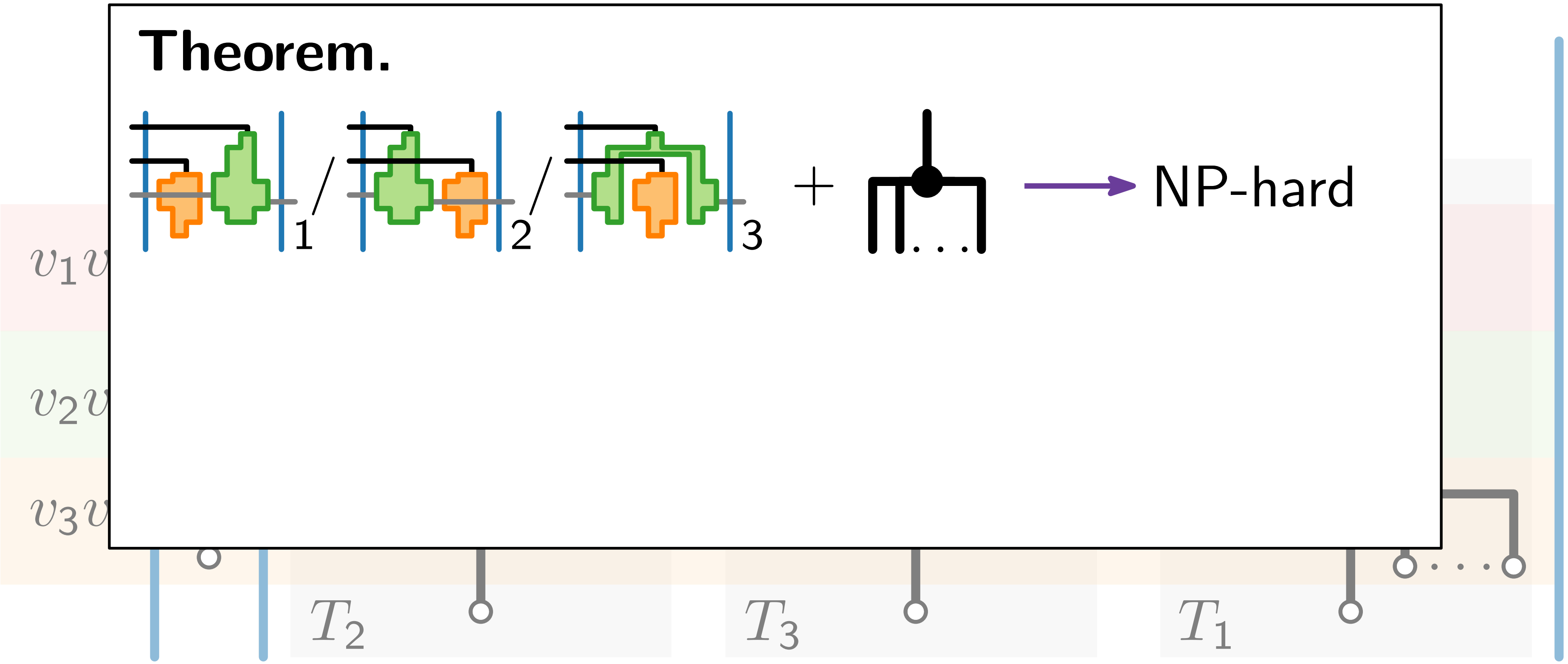




NP-hardness

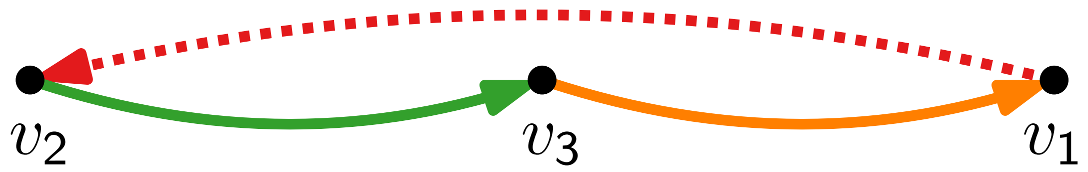
Feedback Arc Set instance

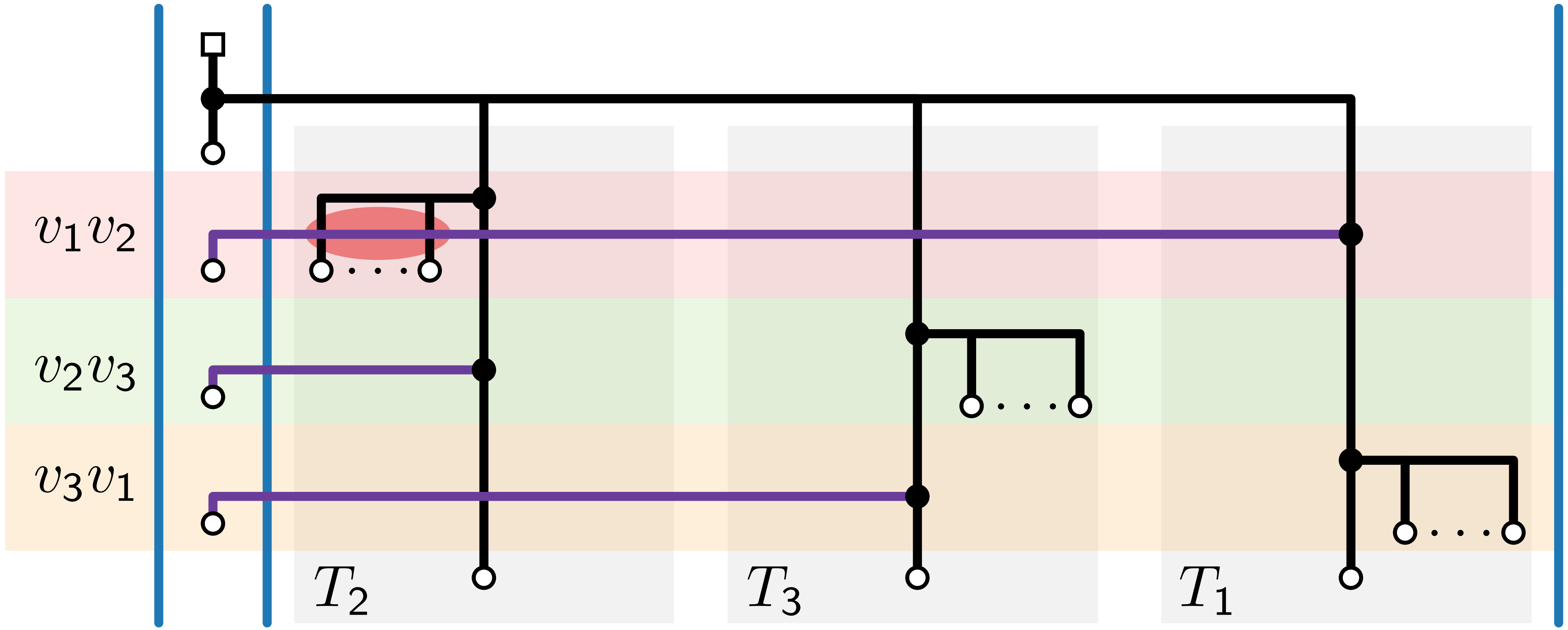




NP-hardness

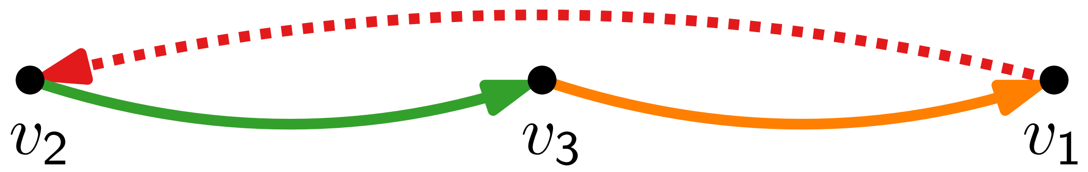
Feedback Arc Set instance

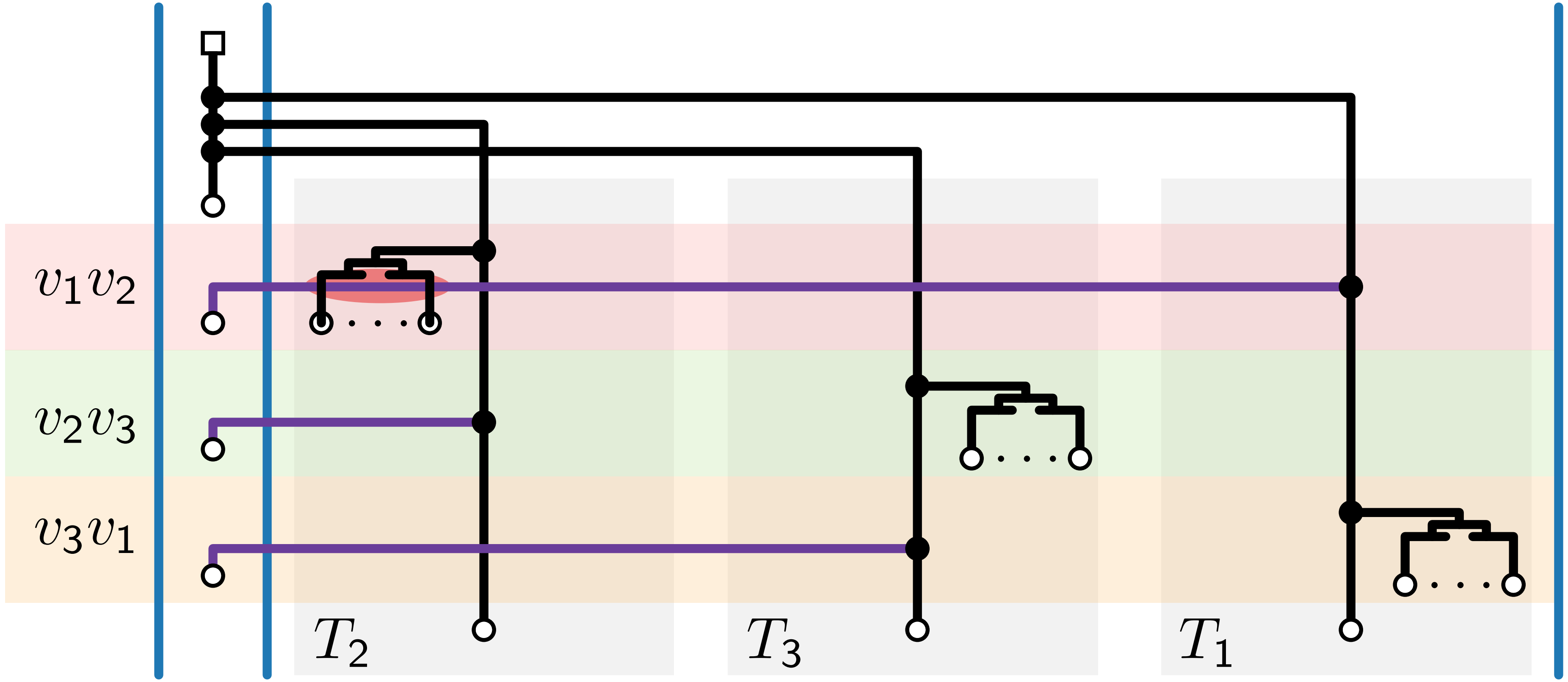




NP-hardness

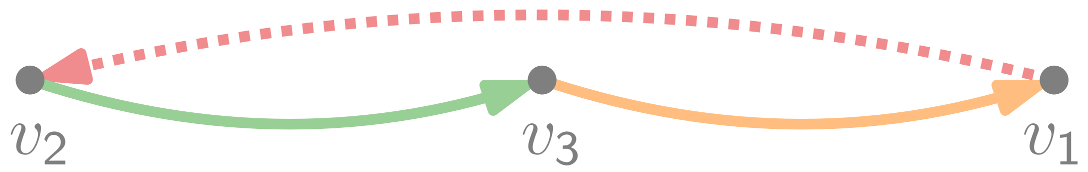
Feedback Arc Set instance

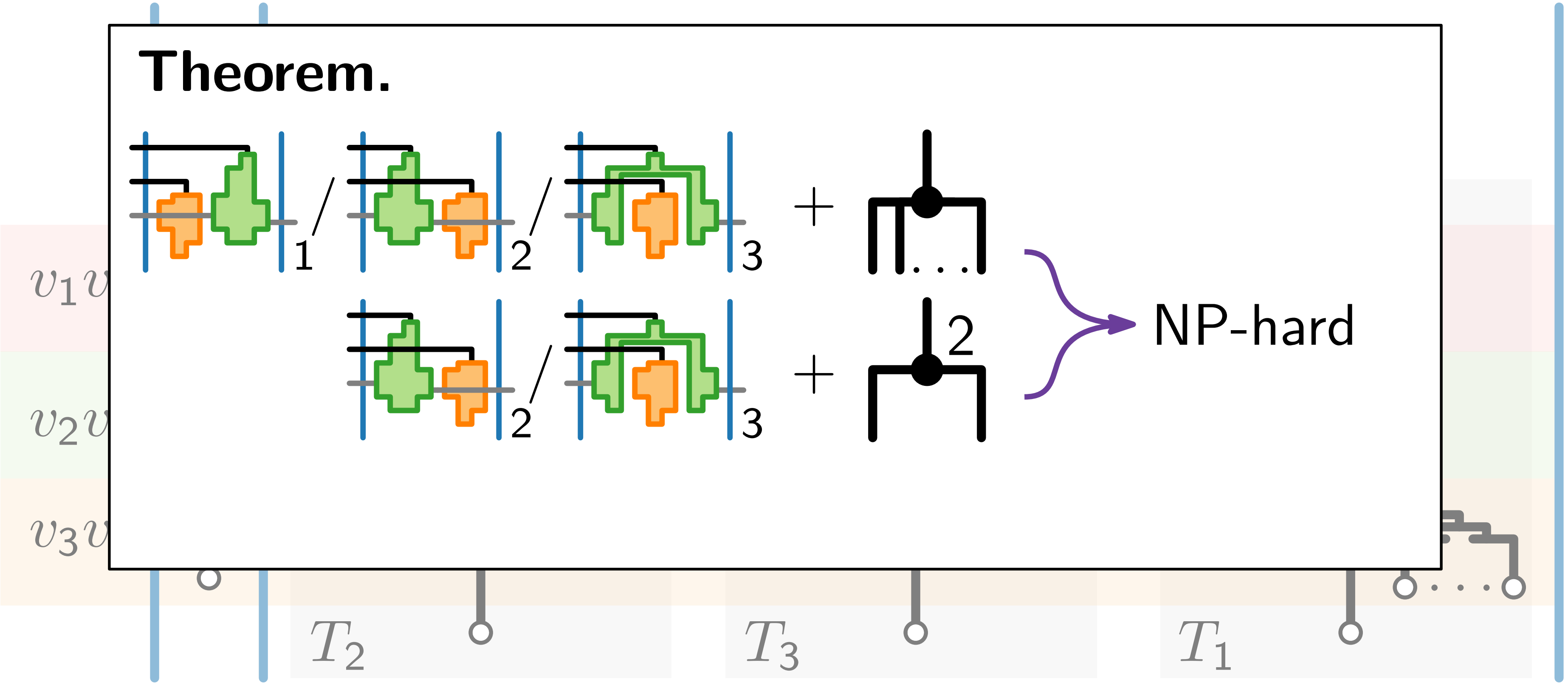




NP-hardness

Feedback Arc Set instance

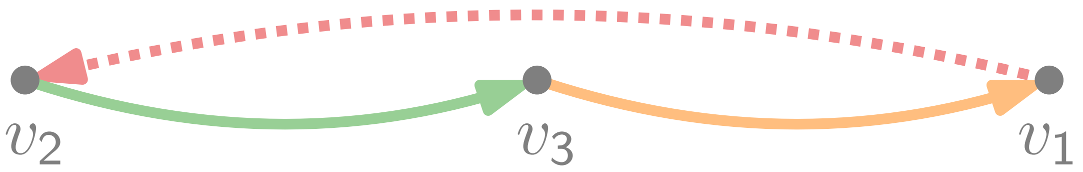




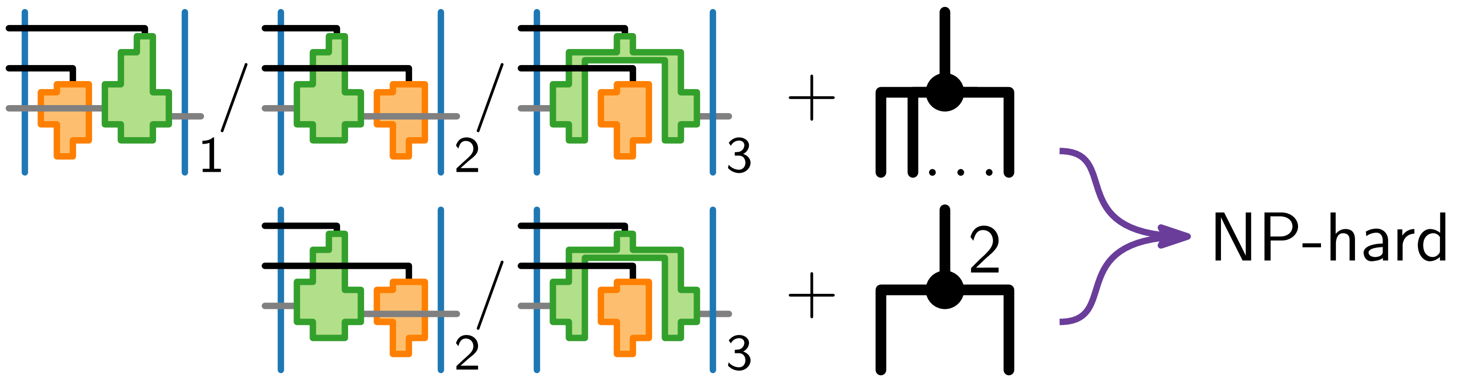
Theorem.

NP-hardness

Feedback Arc Set instance



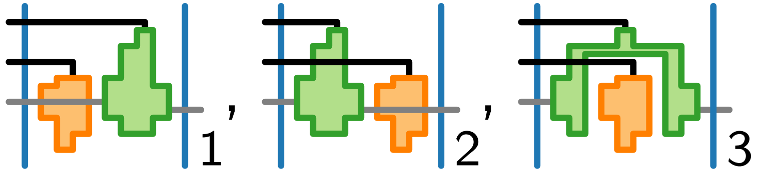
Theorem.



. . . also hard to approximate.

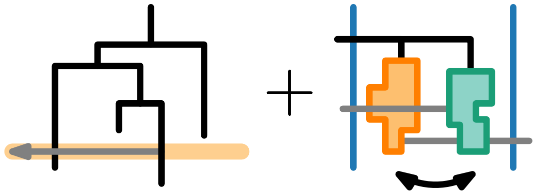
Overview

Drawing Style

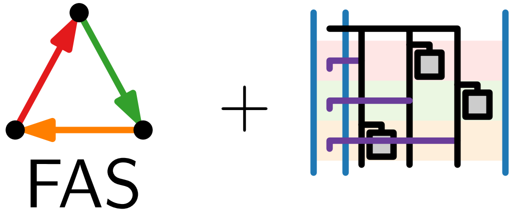


Crossing Minimisation

P



NP



FPT

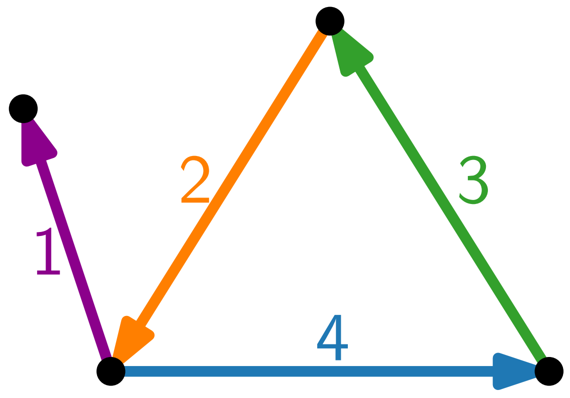
Subtree Arrangement Algorithm

- **subtree arrangement** $\leftrightarrow$ Integer-Weighted FAS (IFAS) $\leftrightarrow$ FAS

Subtree Arrangement Algorithm

- **subtree arrangement** $\leftrightarrow$ Integer-Weighted FAS (IFAS) $\leftrightarrow$ FAS

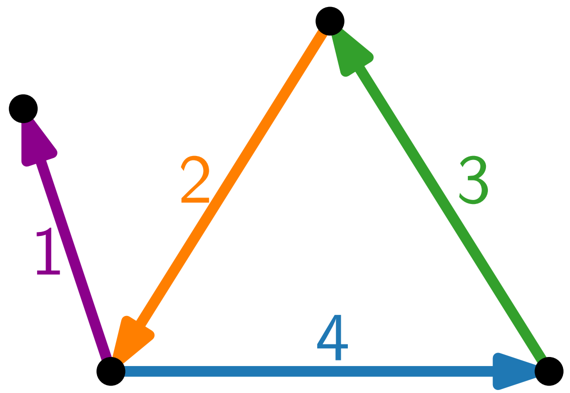
IFAS

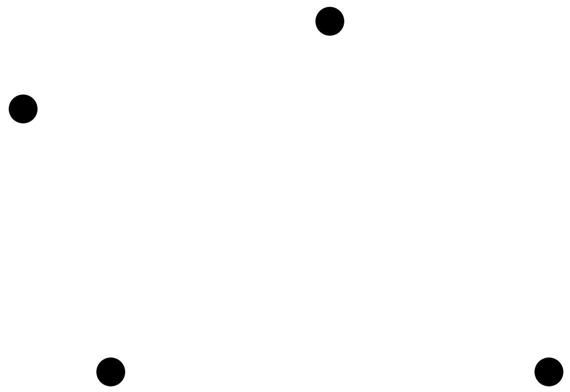


Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS

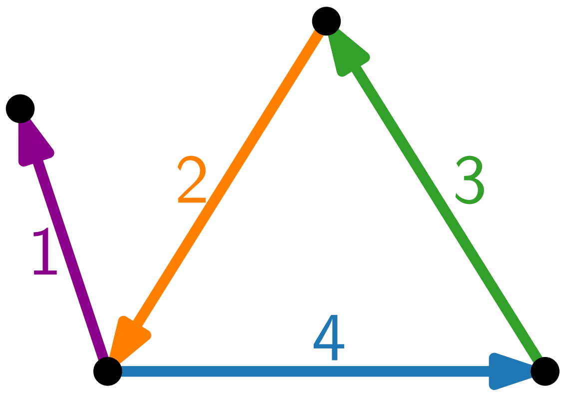


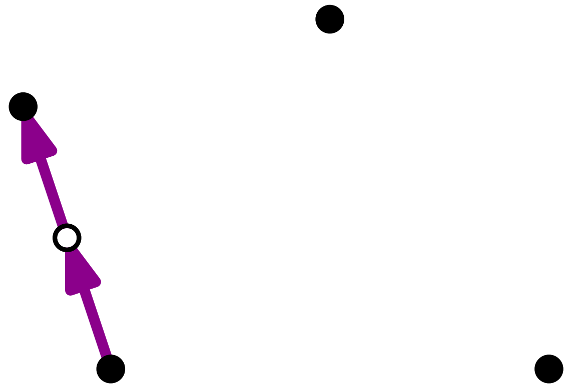


Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS

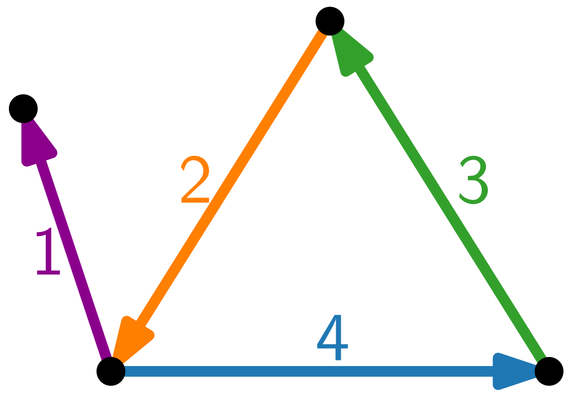


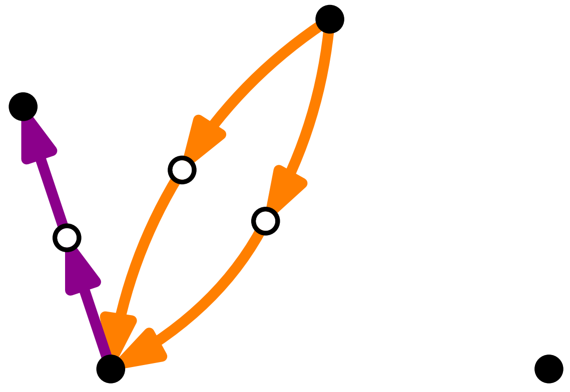


Subtree Arrangement Algorithm

- **subtree arrangement** $\leftrightarrow$ Integer-Weighted FAS (IFAS) $\leftrightarrow$ FAS

IFAS

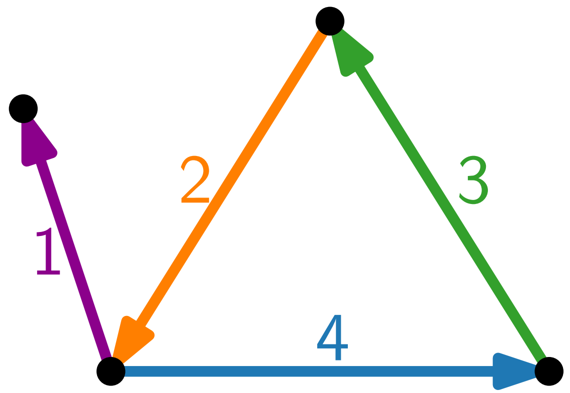


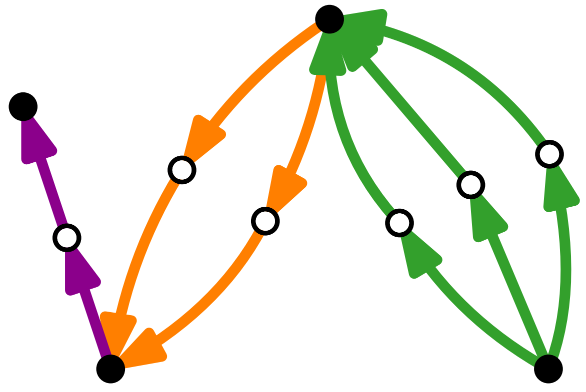


Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS

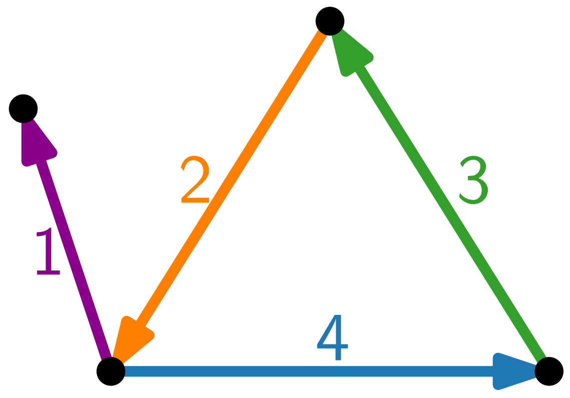


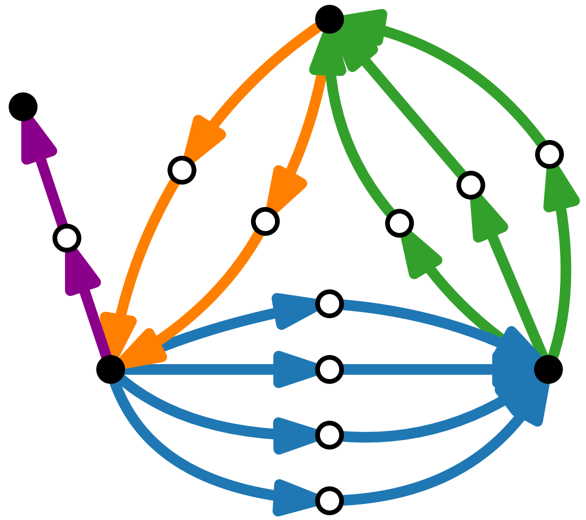


Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS

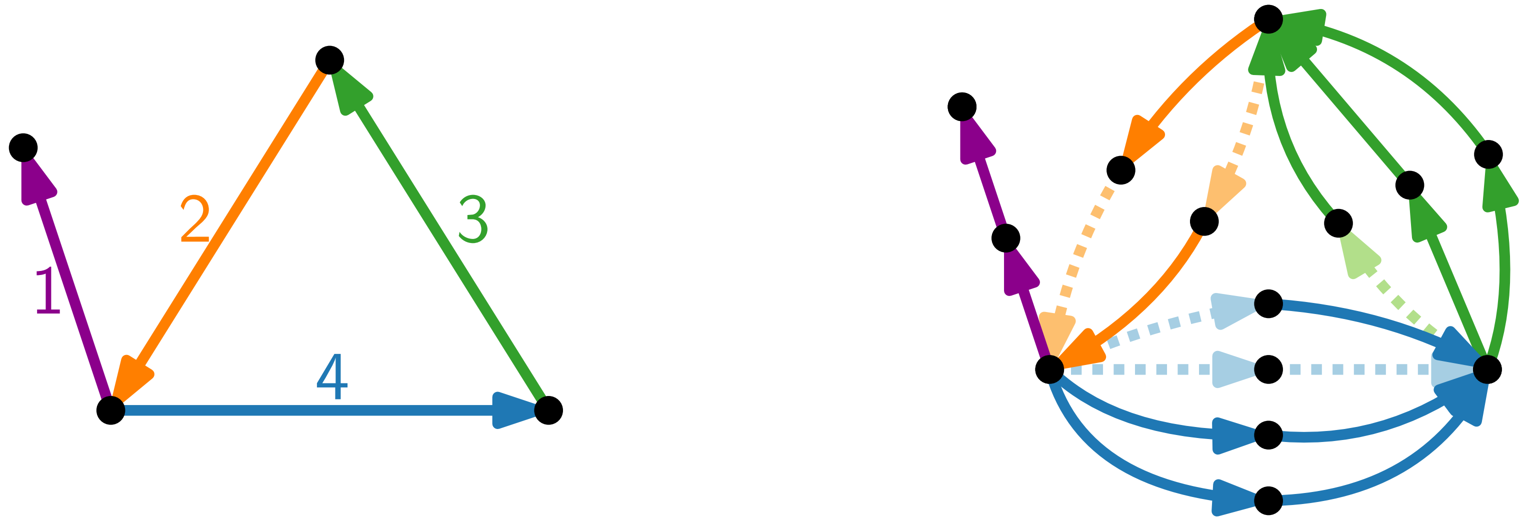




Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

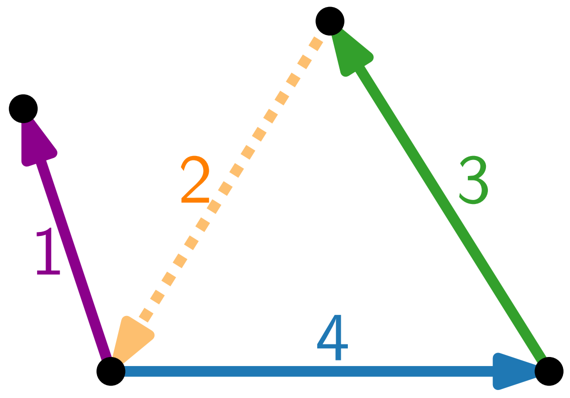
IFAS



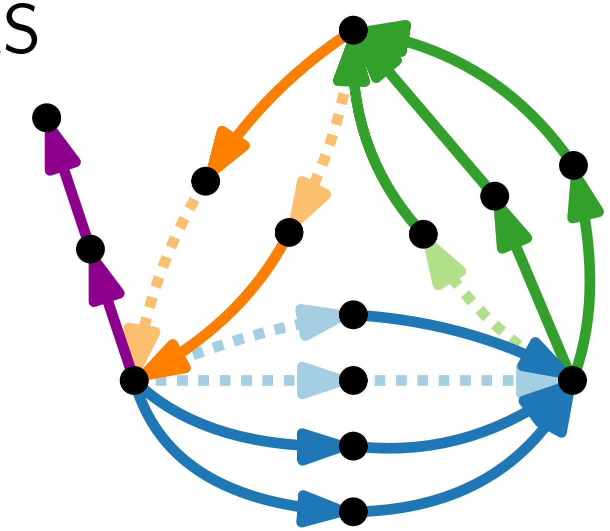
Subtree Arrangement Algorithm

- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS



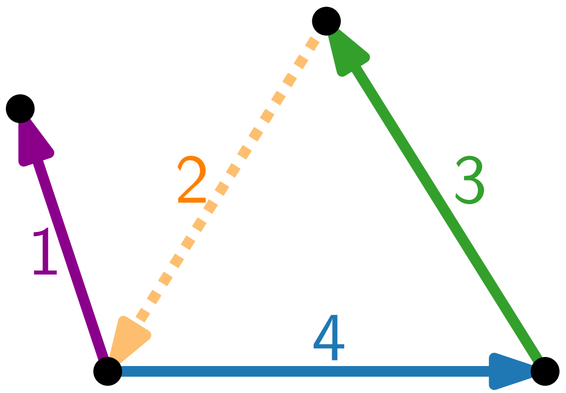
FAS



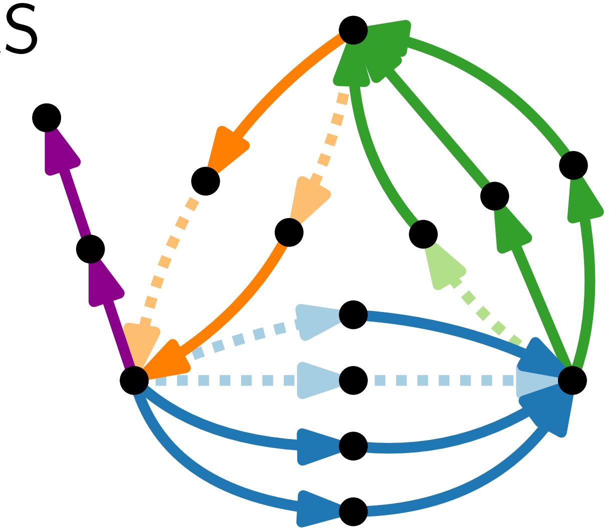
Subtree Arrangement Algorithm

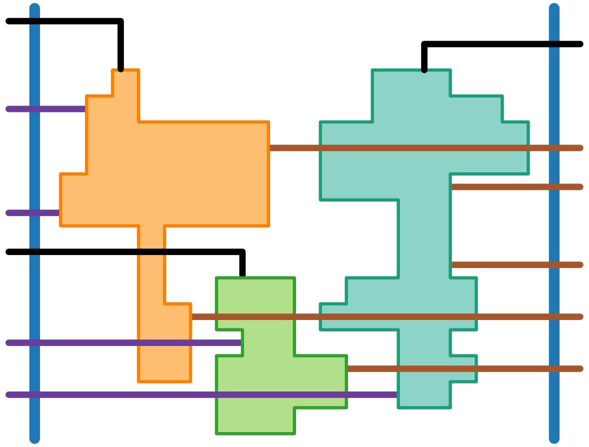
- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS



FAS

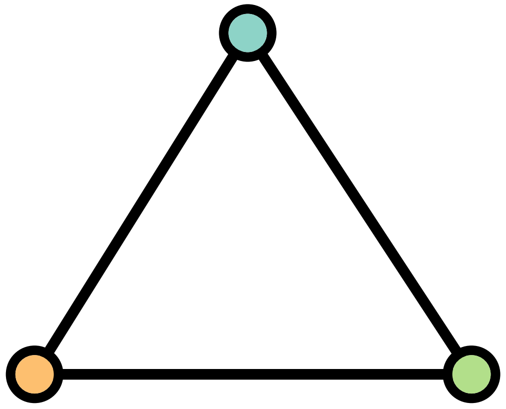




right

left

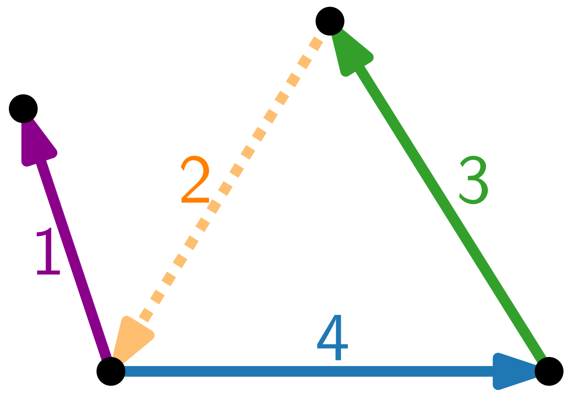
	1	2	3
1	—		
2		—	
3			—



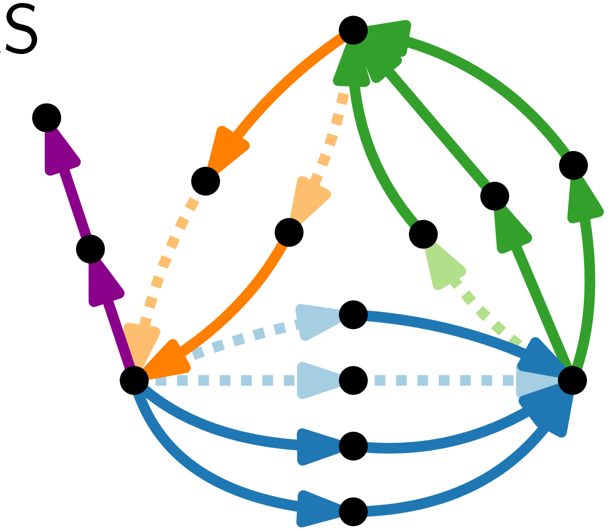
Subtree Arrangement Algorithm

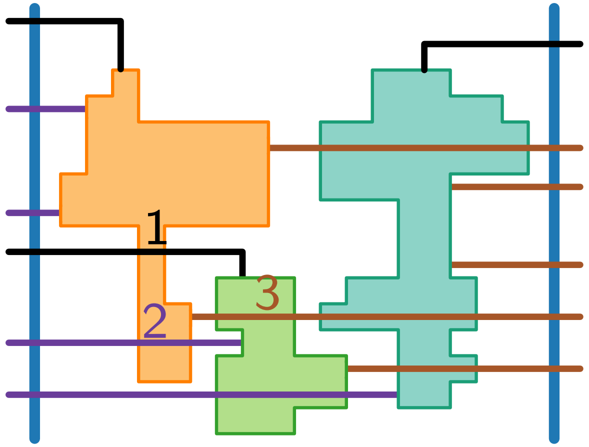
- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

IFAS



FAS

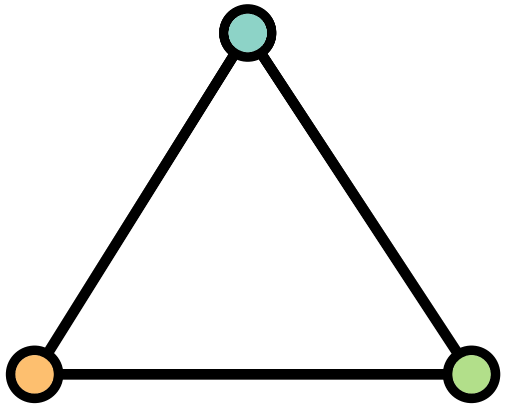




right

left

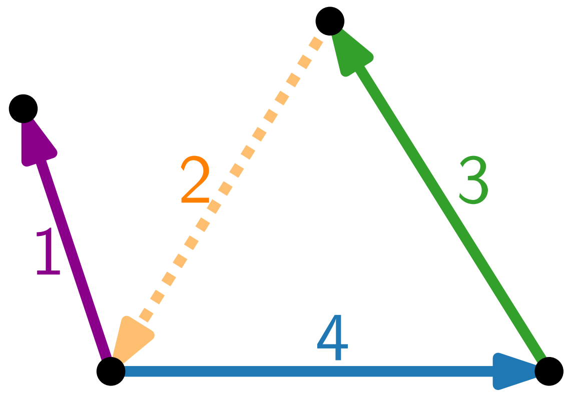
	1	2	3
1	—	6	
2		—	
3			—



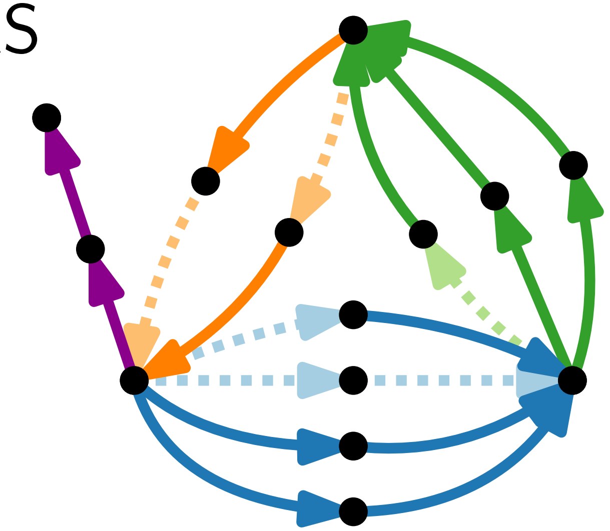
Subtree Arrangement Algorithm

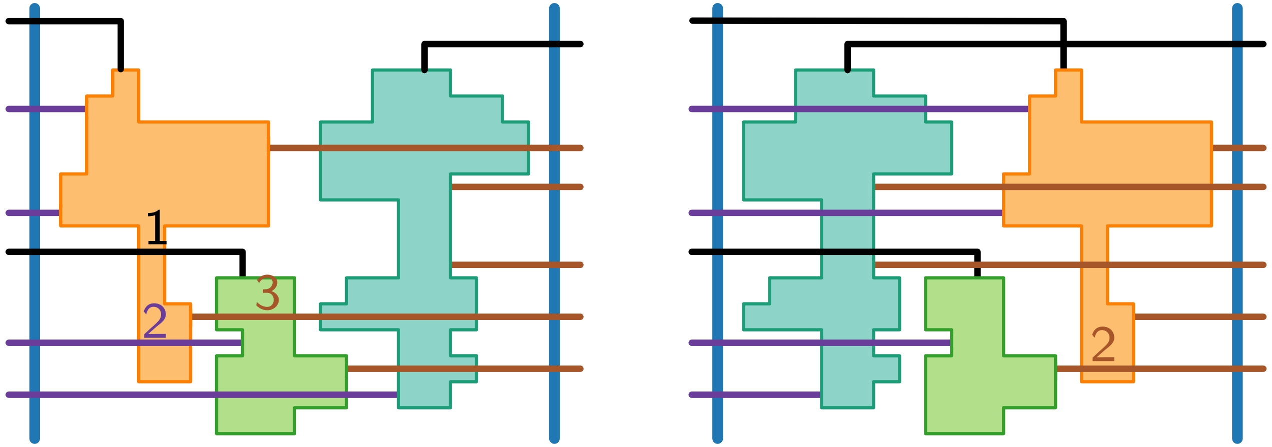
- **subtree arrangement** $\leftrightarrow$ Integer-Weighted FAS (IFAS) $\leftrightarrow$ FAS

IFAS



FAS

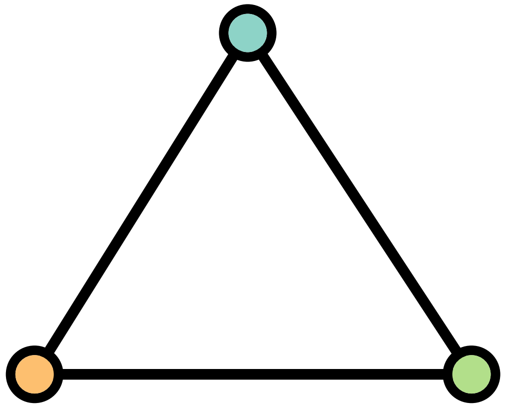




right

left

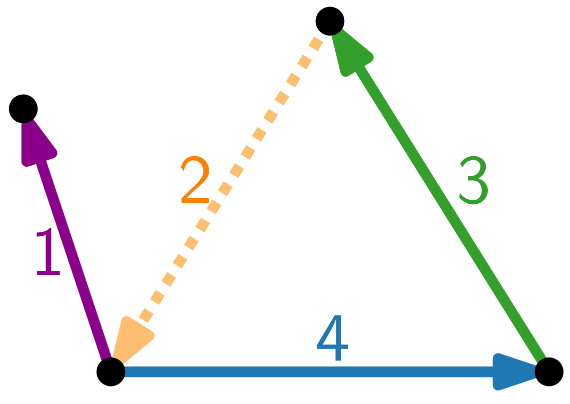
	1	2	3
1	—	6	
2	2	—	
3			—



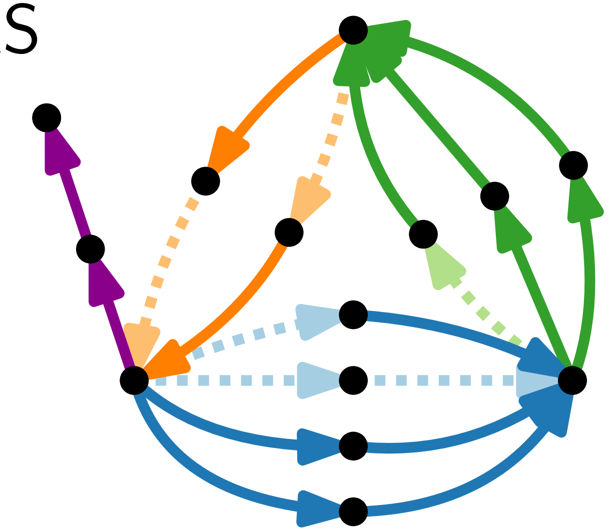
Subtree Arrangement Algorithm

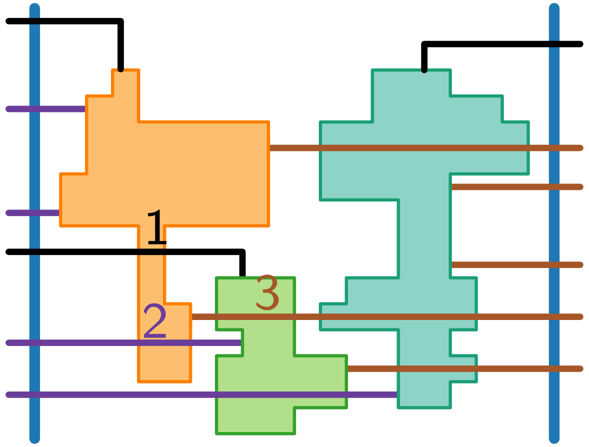
- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

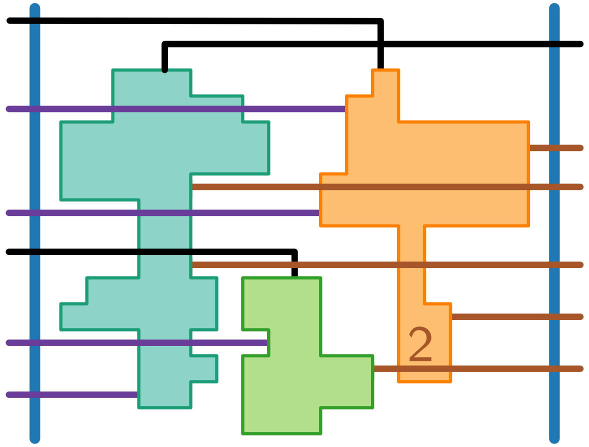
IFAS



FAS

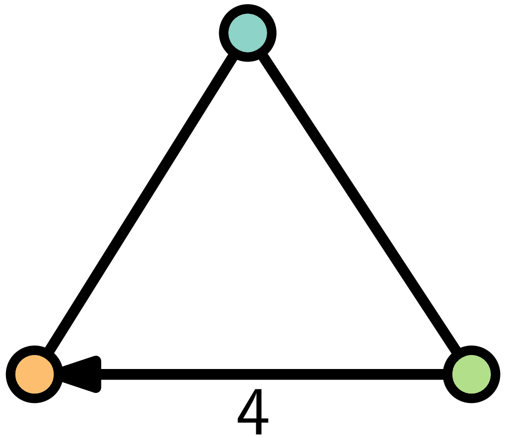






right

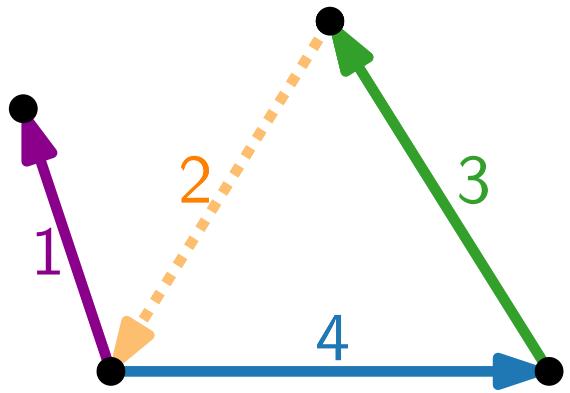
left	1	2	3
1	—	6	
2	2	—	
3			—



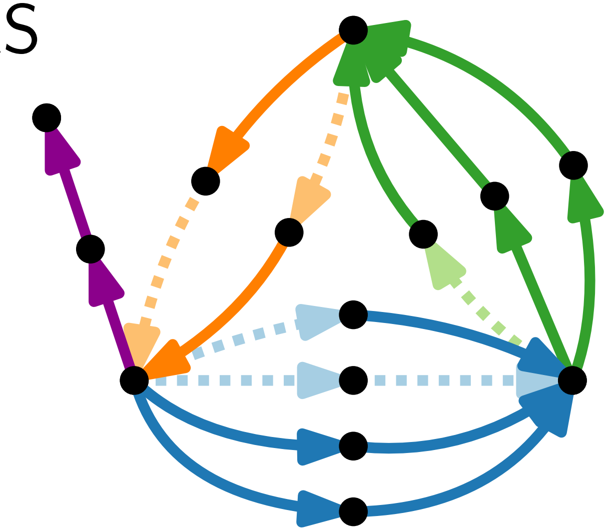
Subtree Arrangement Algorithm

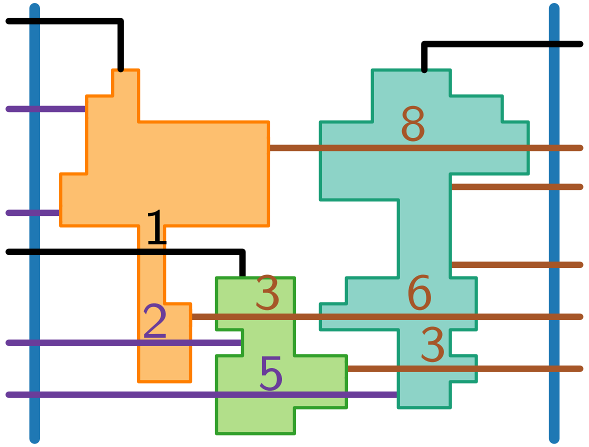
- **subtree arrangement** ↔ Integer-Weighted FAS (IFAS) ↔ FAS

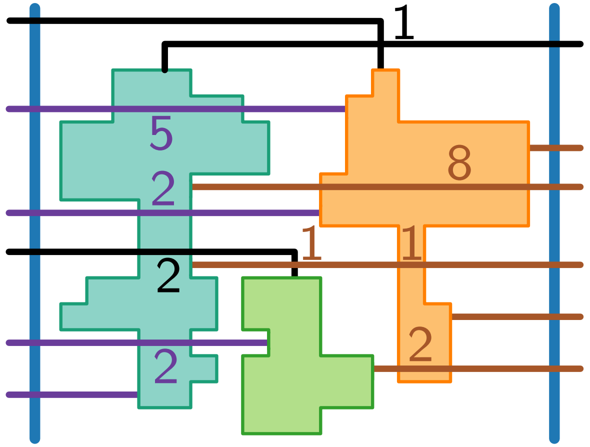
IFAS



FAS



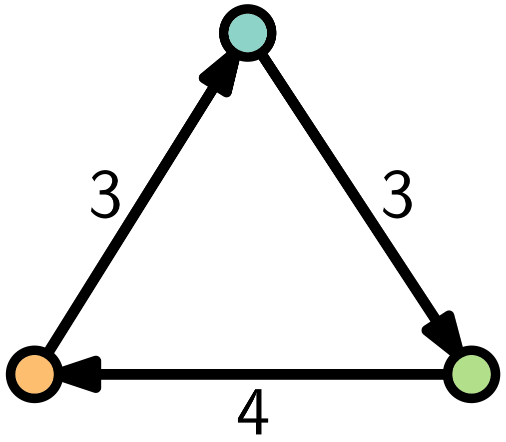




right

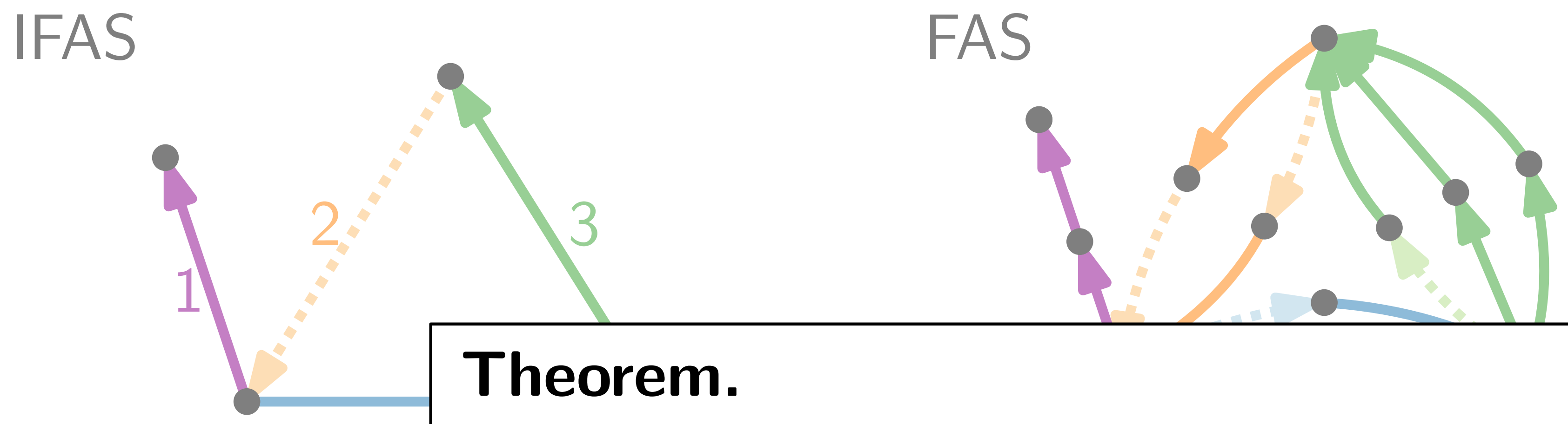
left

	1	2	3
1	—	6	14
2	2	—	8
3	17	5	—

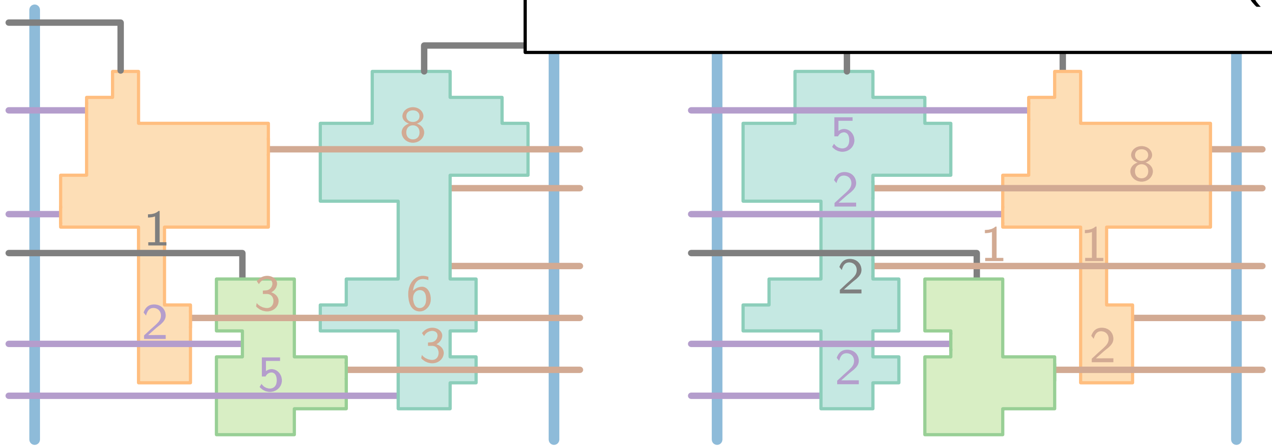


Subtree Arrangement Algorithm

- **subtree arrangement** $\leftrightarrow$ Integer-Weighted FAS (IFAS) $\leftrightarrow$ FAS

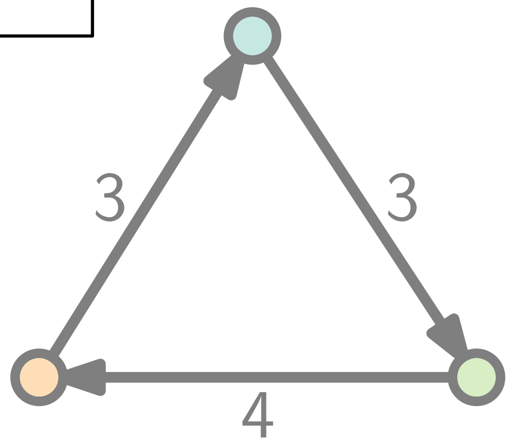


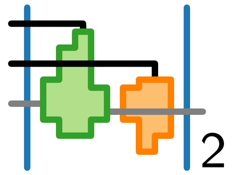
> **Theorem.**
>
> Can reduce our problem in $\mathcal{O}(n^4)$ time to an instance of FAS with $\mathcal{O}(n^4)$ size, ...



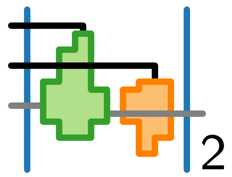
left

	1	2	3
1	–	6	14
2	2	–	8
3	17	5	–

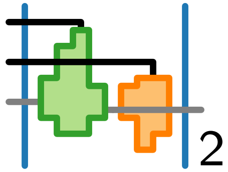


FPT and Approx Algorithms for 

- use **subtree embedding** algo for each maximal subtree in a column
- reduce the **subtree arrangement** problem of each column to a FAS instance

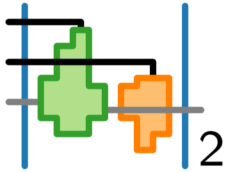
FPT and Approx Algorithms for 

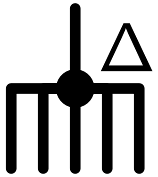
- use **subtree embedding** algo for each maximal subtree in a column
- reduce the **subtree arrangement** problem of each column to a FAS instance
- use FPT and approx algorithms for FAS

FPT and Approx Algorithms for 

- use **subtree embedding** algo for each maximal subtree in a column
- reduce the **subtree arrangement** problem of each column to a FAS instance
- use FPT and approx algorithms for FAS

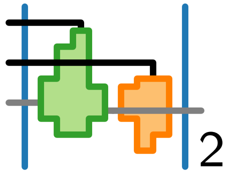
Theorem.



$+$ 

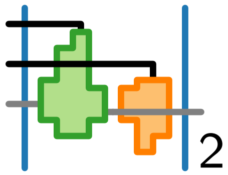
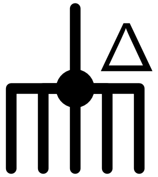
$+$ k crossings

→ FPT algo with running time $\mathcal{O}(\Delta!\Delta n^2 + n^{16} 4^k k^3 k!)$

FPT and Approx Algorithms for 

- use **subtree embedding** algo for each maximal subtree in a column
- reduce the **subtree arrangement** problem of each column to a FAS instance
- use FPT and approx algorithms for FAS

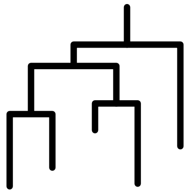
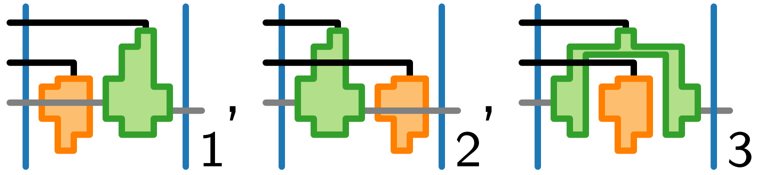
Theorem.

 +  + k crossings

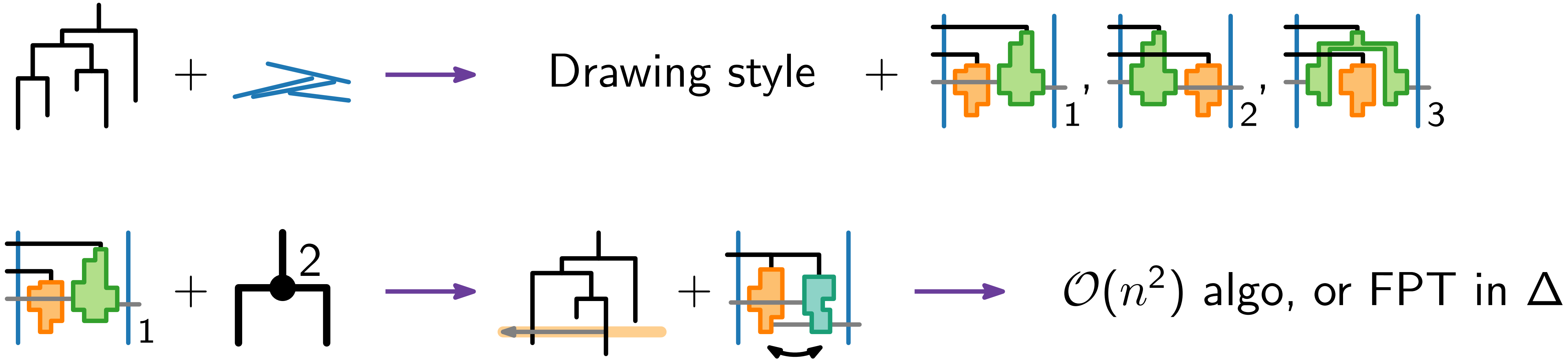
→ FPT algo with running time $\mathcal{O}(\Delta!\Delta n^2 + n^{16} 4^k k^3 k!)$

→ $\mathcal{O}(\text{poly}(n)\Delta!\Delta)$-time approx algo with approx factor in $\mathcal{O}(\log n \log\log n)$

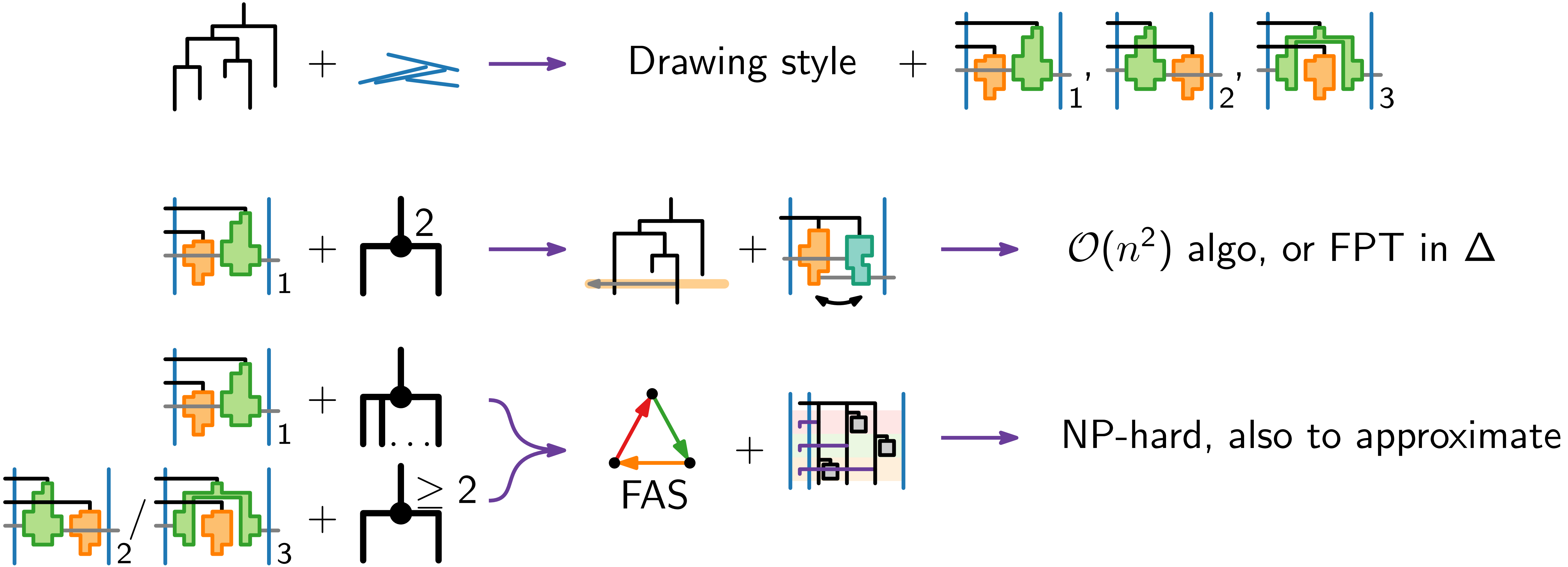
Summary

 +  → Drawing style + 

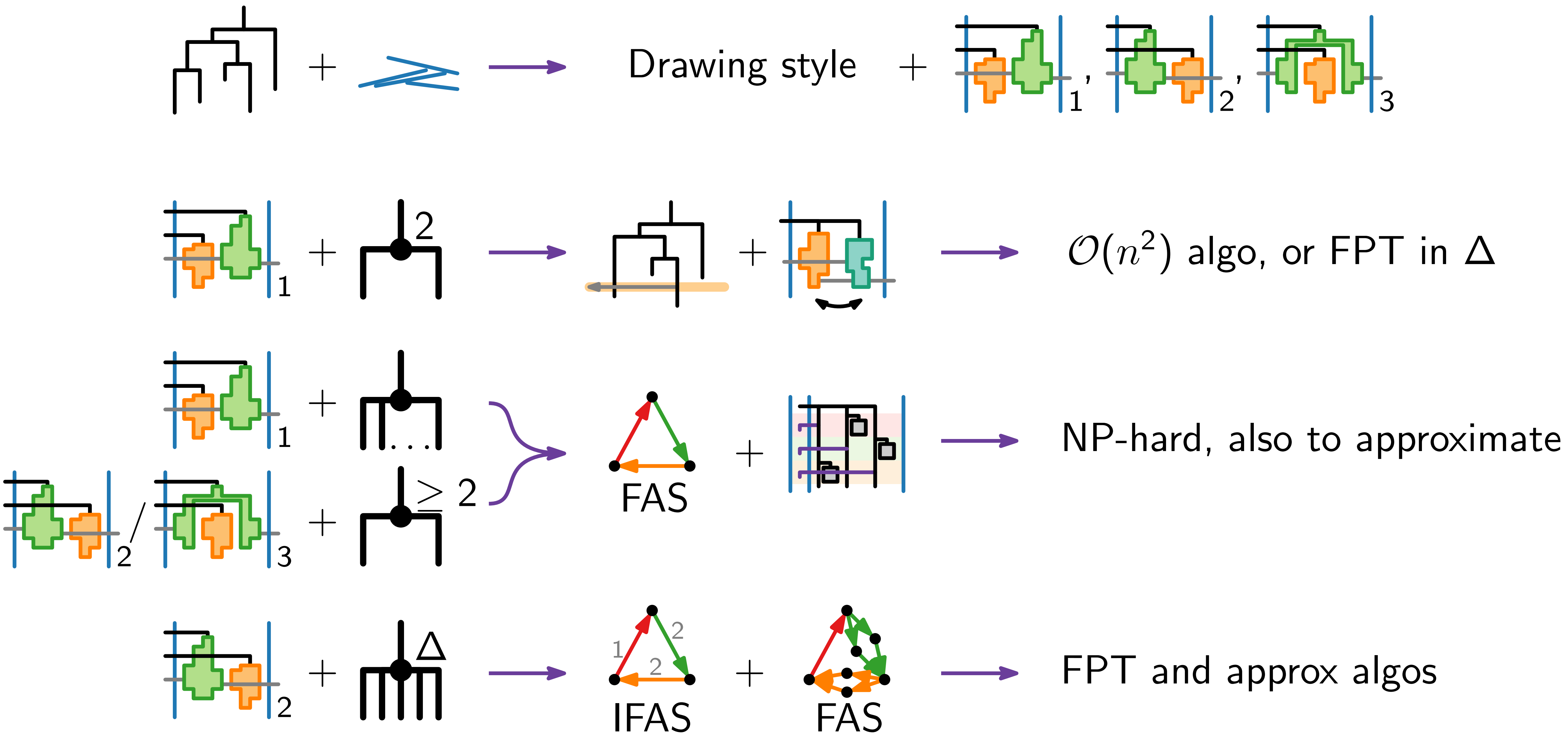
Summary



Summary


+
Drawing style
+
1, 2, 3
1 + 2
+
$\mathcal{O}(n^2)$ algo, or FPT in Δ
1 + . . .
2 / 3 + ≥ 2
FAS
+
NP-hard, also to approximate

Summary



Drawing style
1, 2, 3
2
O(n²) algo, or FPT in Δ
1
2 / 3
≥ 2
FAS
NP-hard, also to approximate
2
Δ
1 2 2
IFAS
FAS
FPT and approx algos