

# Parameterized Algorithms

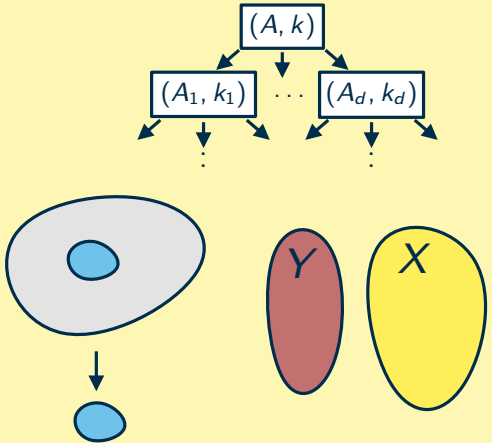
## Kernelization: Similar Trees

Thomas Bläsius

# Content

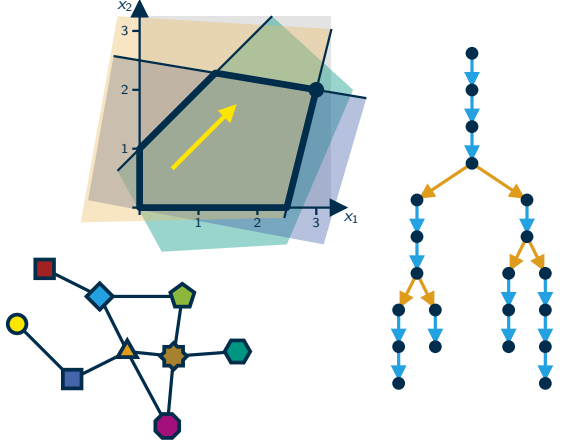
## Basic toolbox

- bounded search trees
- kernelization
- iterative compression



## Extended toolbox

- linear programs
- branch-and-reduce
- color coding



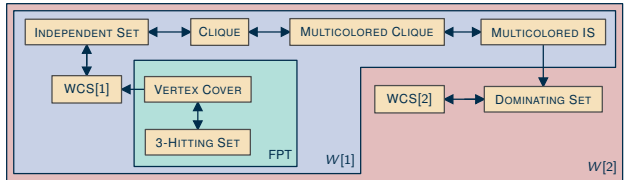
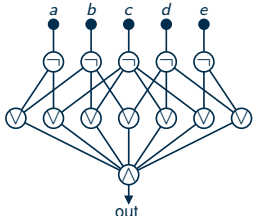
## Tree width

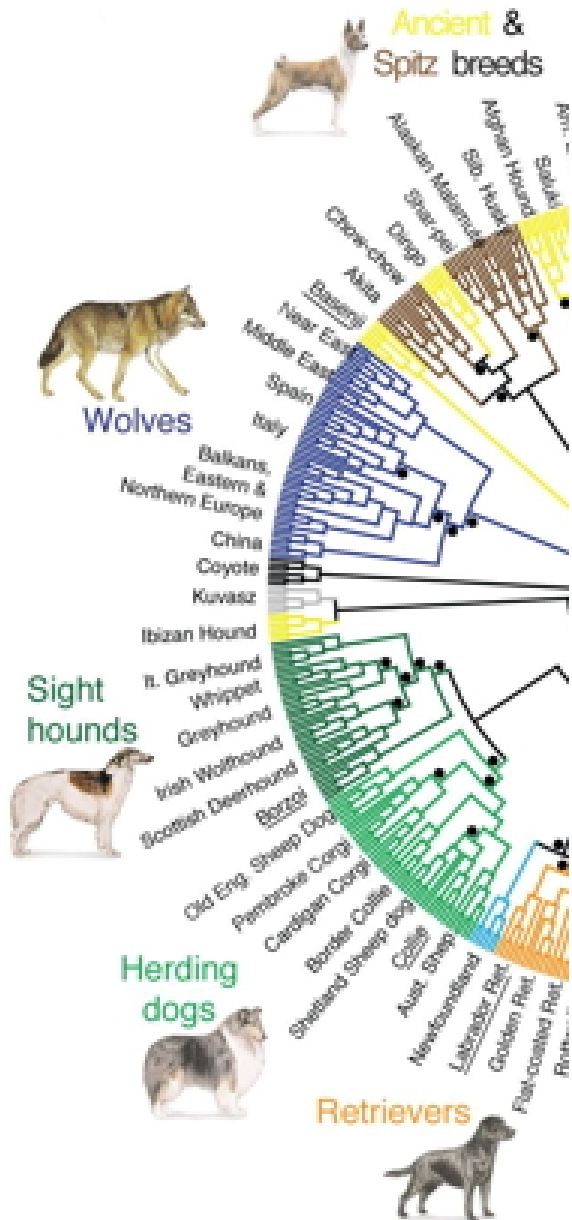
- dynamic programming
- chordal and planar graphs
- Courcelle's theorem



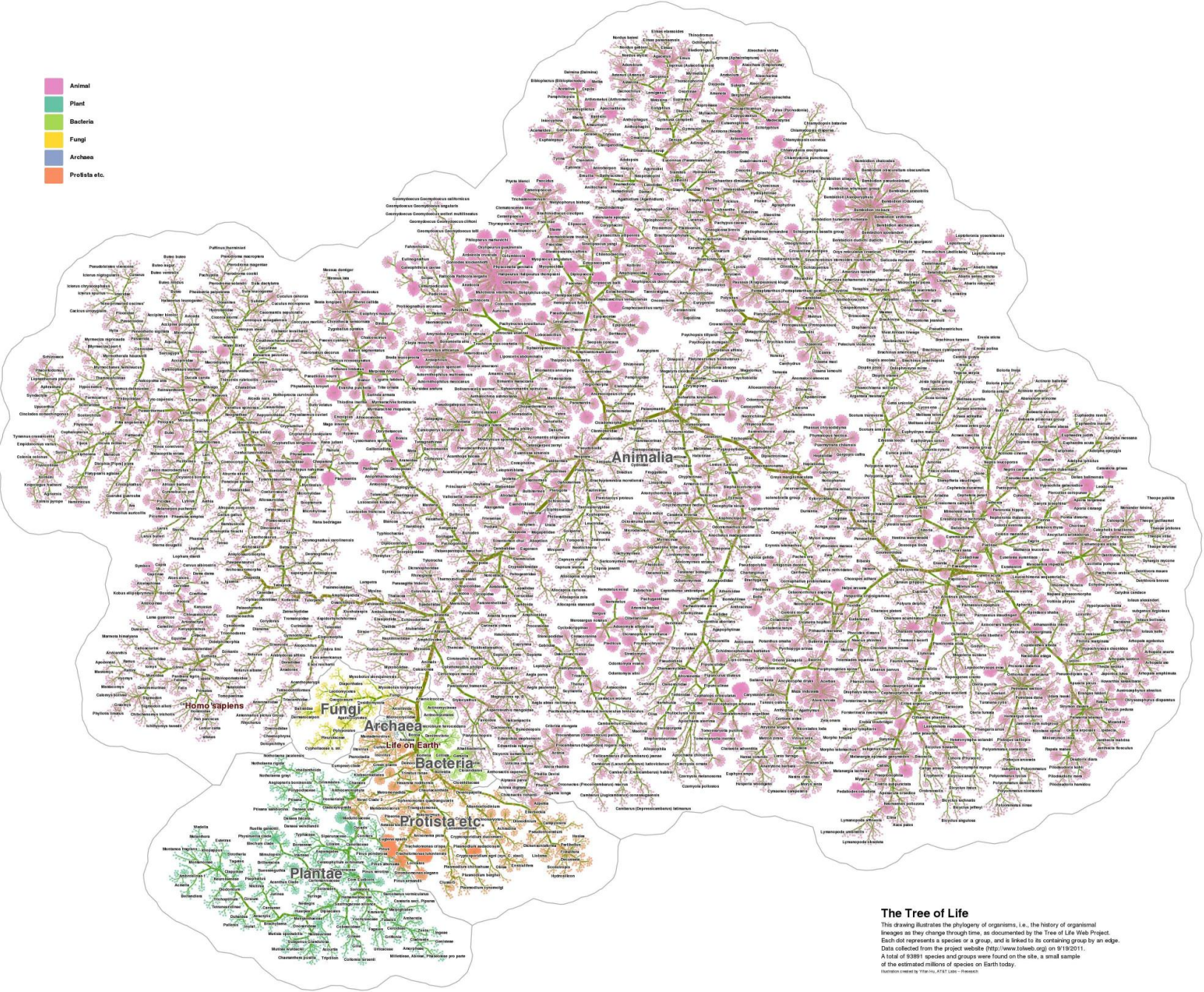
## Lower bounds

- kernel lower bounds
- parameterized reductions
- circuits and the W-hierarchy
- ETH and SETH





- Animal
- Plant
- Bacteria
- Fungi
- Archaea
- Protista etc.



**The Tree of Life**  
 This drawing illustrates the phylogeny of organisms, i.e., the history of organismal lineages as they change through time, as documented by the Tree of Life Web Project. Each dot represents a species or a group, and is linked to its containing group by an edge. Data collected from the project website (<http://www.tolweb.org>) on 9/19/2011. A total of 93981 species and groups were found on the site, a small sample of the estimated millions of species on Earth today.  
 Illustration created by Wiley (LH), AT&T Labs - Research

# Phylogenetic trees

## Definition

A **phylogenetic tree** is an unrooted full binary tree, such that each leaf has a unique label.

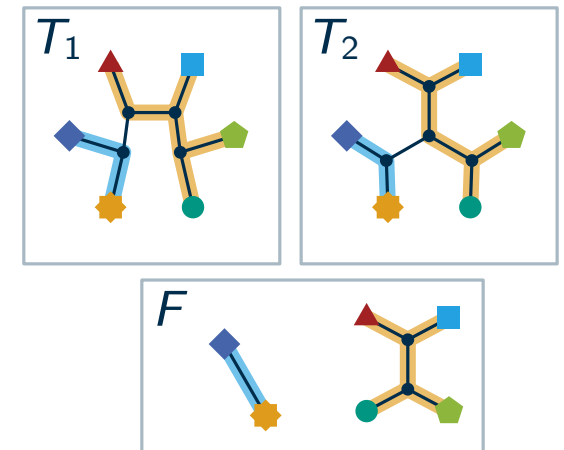
→ are often generated automatically, e.g., based on DNA sequencing

## Problem

- different algorithms yield different trees
- different data (e.g., due to measuring errors) yield different trees
- How to compare different trees  $T_1$  and  $T_2$  on the same set of leaves  $L$ ?

## Maximum agreement forest

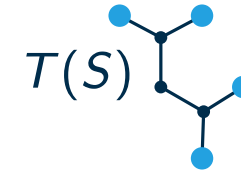
- forest  $F$  of full binary trees with leaves  $L$
- trees in  $F$  are contained in  $T_1$  and  $T_2$  (as minors)
- minimize the number of trees in  $F$



# What does this mean exactly?

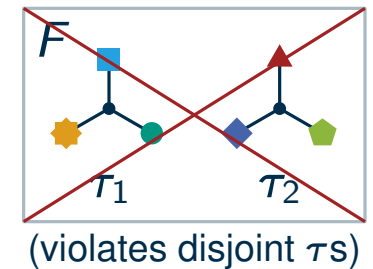
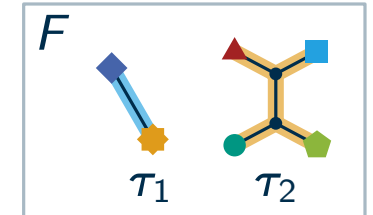
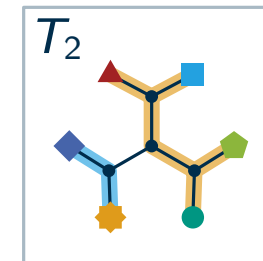
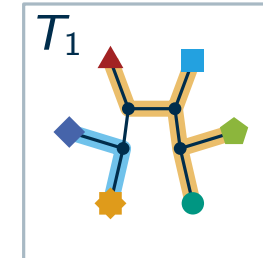
## Leaf-induced subtrees

- let  $T$  be a tree with leaves  $L(T)$  and let  $S \subseteq L(T)$
- $T(S) =$  minimal subtree of  $T$  that contains  $S$
- contract all vertices of degree 2 in  $T(S) \rightarrow T[S]$



## Agreement forest

- let  $F$  be a forest consisting of the trees  $\tau_1, \dots, \tau_k$
- $T_1$  and  $T_2$  are trees (same leaves  $L(T_1) = L(T_2) = L(F)$ )
- $F$  is an **agreement forest** of  $T_1$  and  $T_2$ , if:
  - each  $\tau_i$  included in  $T_1$  and  $T_2$ :  $T_1[L(\tau_i)] = T_2[L(\tau_i)] = \tau_i$
  - disjoint  $\tau$ s:  $T(L(\tau_i)) \cap T(L(\tau_j)) = \emptyset$  (for  $T \in \{T_1, T_2\}$ )



## Problem: MAXIMUM AGREEMENT FOREST

Given  $T_1, T_2$ , and a parameter  $k$ . Is there an agreement forest of at most  $k$  trees?

# Plan: Kernelization

## Problem: MAXIMUM AGREEMENT FOREST

Given  $T_1, T_2$ , and a parameter  $k$ . Is there an agreement forest of at most  $k$  trees?

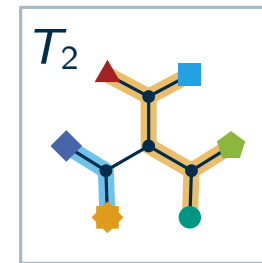
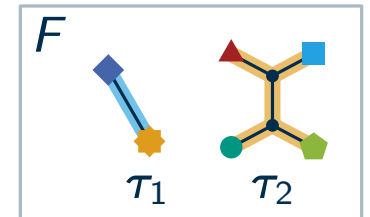
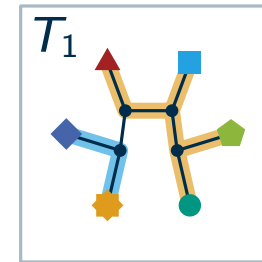
### Goal

- find a kernelization algorithm  $\rightarrow$  polynomial reduction rules
- size of the resulting trees only depends on  $k$

### Before we start reducing: What is the goal?

- size of the resulting trees only depends on  $k$
- slightly stronger but more specific claim:
  - every tree in the agreement forest has few leaves
  - implies: few leaves overall  $\Rightarrow$  small instance
- Which tree structures contradict this claim? Can we remove them using reduction rules?

Suggestions?



# Undesirable tree structure (1)

**Claim:** Every tree of the agreement forest has only few leaves.

## Counterexample

- $T_1$  and  $T_2$  may share a subtree  $T'$  with many leaves
- in the agreement forest,  $T'$  could be one of the  $\tau_1, \dots, \tau_k$

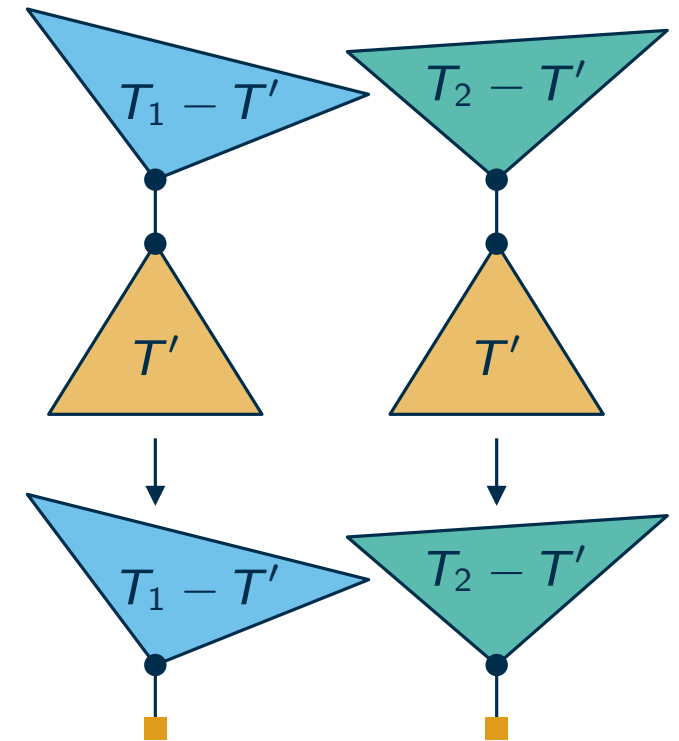
## Reduction rule 1

- find edges in  $T_1$  and  $T_2$  that cut off the same subtree  $T'$
- replace  $T'$  by a leaf with a new label (same label in  $T_1$  and  $T_2$ )

## Lemma

Reduction rule 1 is safe and can be applied in polynomial time.

**Proof:** next slide



# Reduction rule 1

## Reduction rule 1

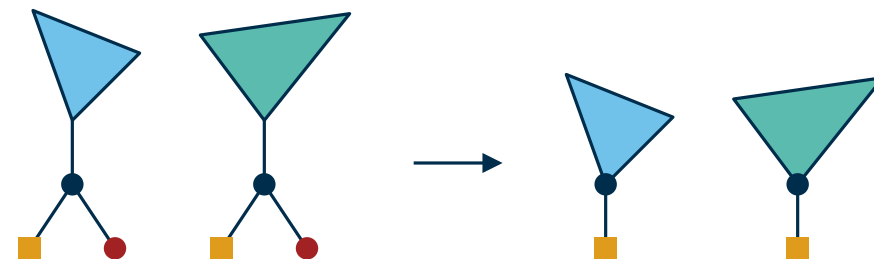
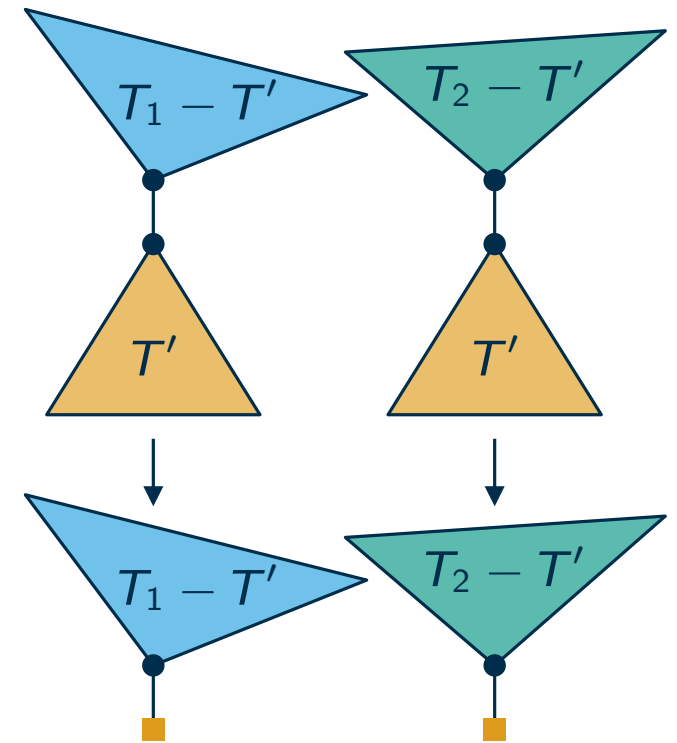
- find edges in  $T_1$  and  $T_2$  that cut off the same subtree  $T'$
- replace  $T'$  by a leaf with a new label (same label in  $T_1$  and  $T_2$ )

### Lemma

Reduction rule 1 is safe and can be applied in polynomial time.

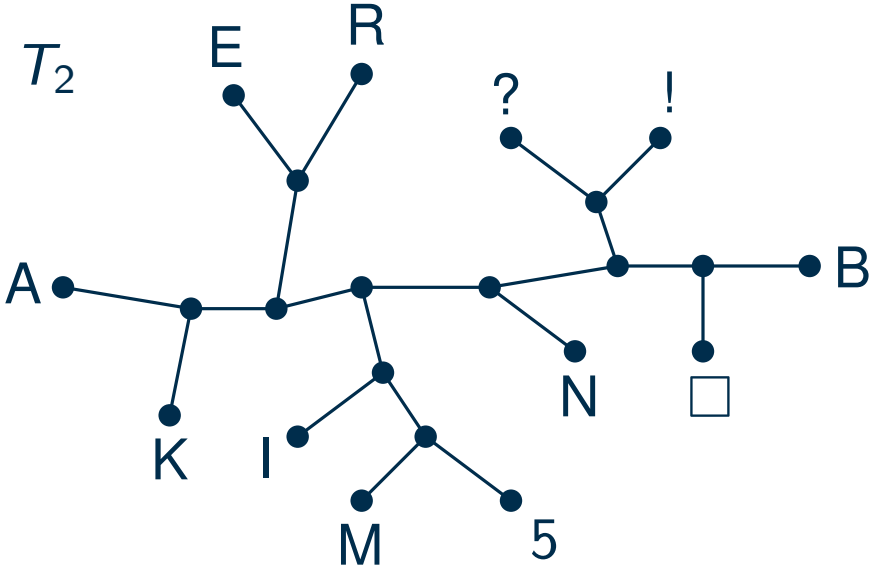
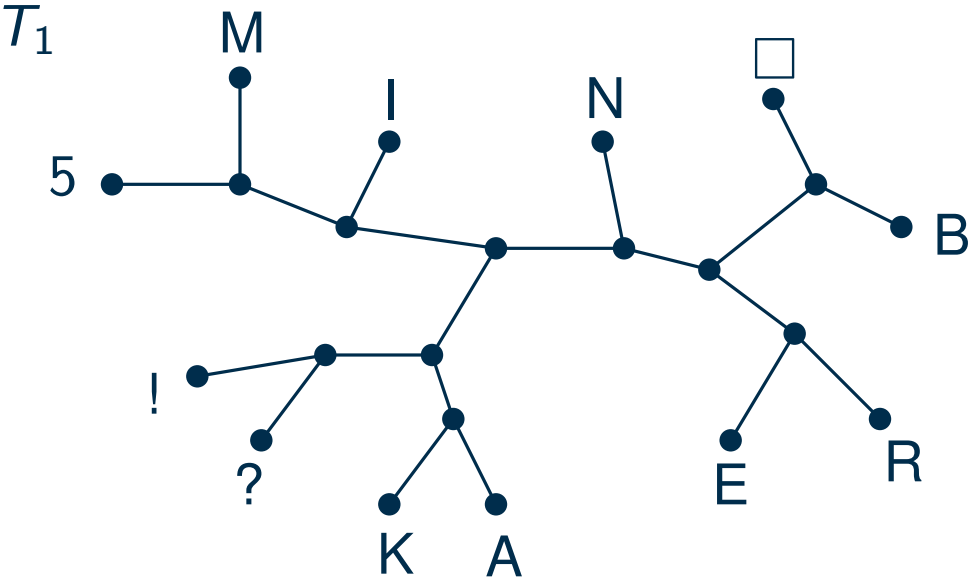
### Proof

- let  $F'$  be an agreement forest of the new instance and let  $\tau'_1$  be the tree with the new leaf in  $F'$
- replace this leaf in  $\tau'_1$  by  $T'$   $\rightarrow$  agreement forest of the same size for  $T_1$  and  $T_2$
- other direction: similar
- polynomial running time: for example by iteratively eating “cherries”



# Example

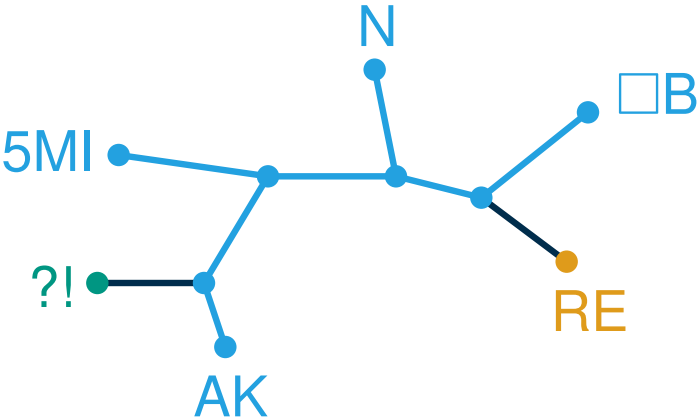
Solve this instance of **MAXIMUM AGREEMENT FOREST**



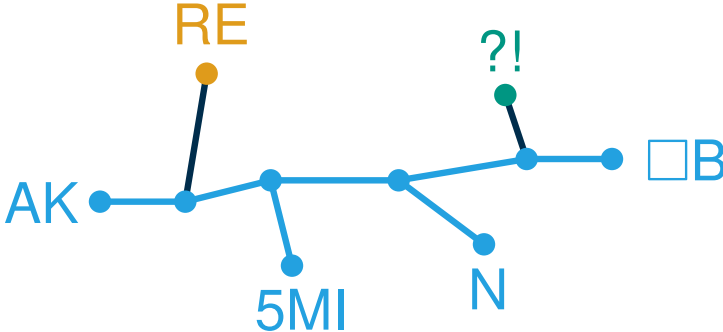
# Example

Solve this instance of **MAXIMUM AGREEMENT FOREST**

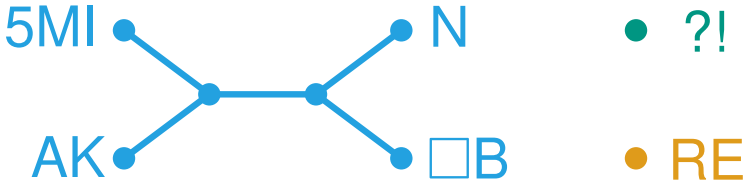
$T_1$



$T_2$



$F$



$\Rightarrow$  solution has size 3

# Undesirable tree structures 2

**Claim:** Every tree of the agreement forest has only few leaves. (after applying reduction rule 1)

## Counterexample

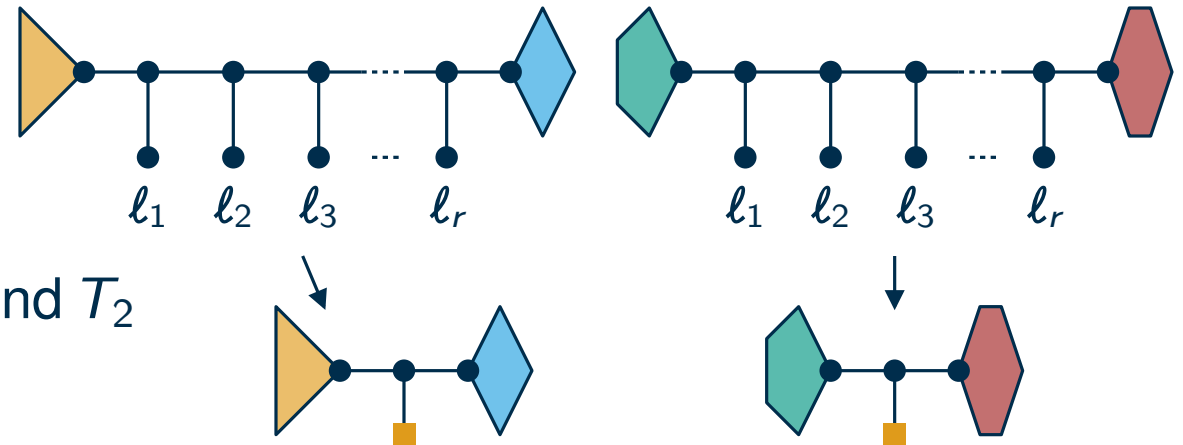
- many leaves in a shared subtree

## Reduction rule 2

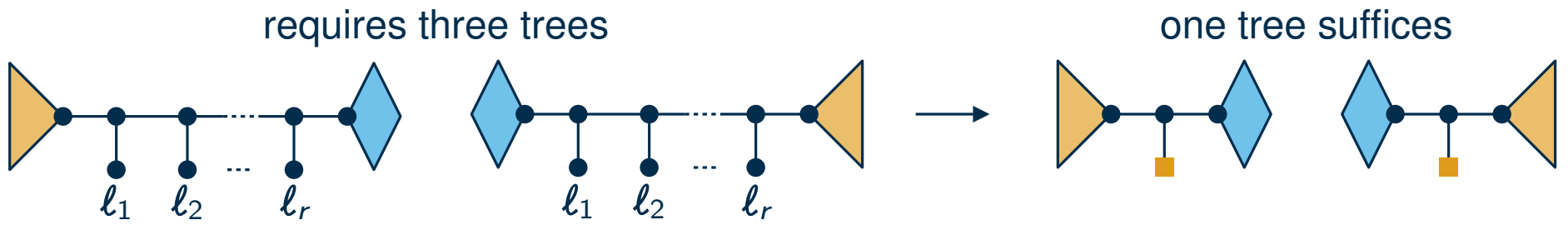
- find path with attached leaves  $l_1, \dots, l_r$  in  $T_1$  and  $T_2$
- replace the path by a vertex with a new leaf

## Problem

- number of required trees may be reduced



Is this rule safe?



# Undesirable tree structures 2

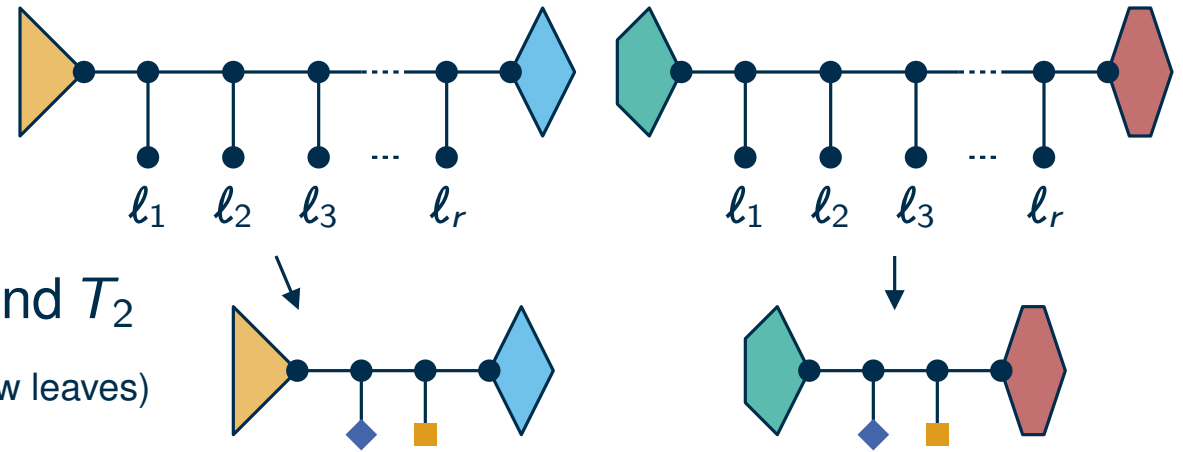
**Claim:** Every tree of the agreement forest has only few leaves. (after applying reduction rule 1)

## Counterexample

- many leaves in a shared subtree

## Reduction rule 2

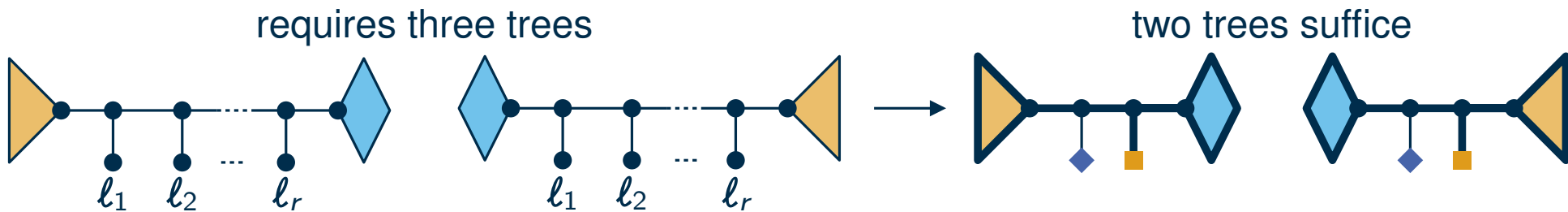
- find path with attached leaves  $l_1, \dots, l_r$  in  $T_1$  and  $T_2$
- replace it with a path of length two (two new leaves)



Is this rule safe?

## Problem

- number of required trees may be reduced



# Undesirable tree structures 2

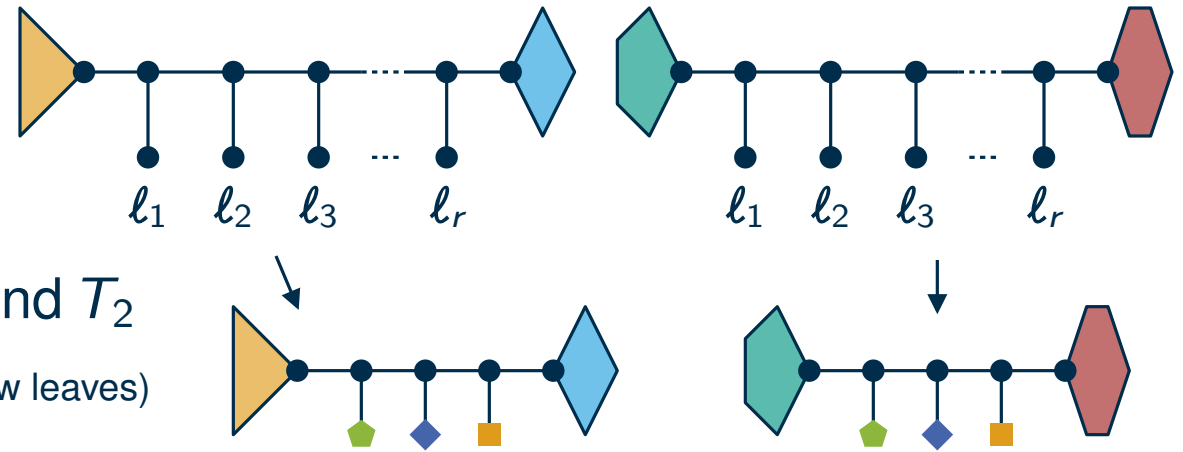
**Claim:** Every tree of the agreement forest has only few leaves. (after applying reduction rule 1)

## Counterexample

- many leaves in a shared subtree

## Reduction rule 2

- find path with attached leaves  $l_1, \dots, l_r$  in  $T_1$  and  $T_2$
- replace it with a path of length three (three new leaves)



**Lemma:** Reduction rule 2 is safe and can be applied in polynomial time.

## Proof

- case distinction (not here; but not very difficult)
- polynomial running time: for example by iteratively shortening paths of length 4

# Undesirable tree structures 3

amortized

**Claim:** Every tree of the agreement forest has only few leaves. (after applying rules 1 and 2)

## Counterexample

- many leaves in a shared subtree



## Amortized perspective

- after every three leaves, we see different subtrees
- each of them yields at least one tree in the agreement forest
- few trees in the agreement forest  $\Rightarrow$  few leaves

Is this really a problem?

(if  $a, b, c, d, \dots$  are all in the same tree)

## Lemma

Exhaustively applying reduction rules 1 and 2 to a yes-instance yields an equivalent instance with at most  $ck$  leaves (for a small constant  $c$ ).

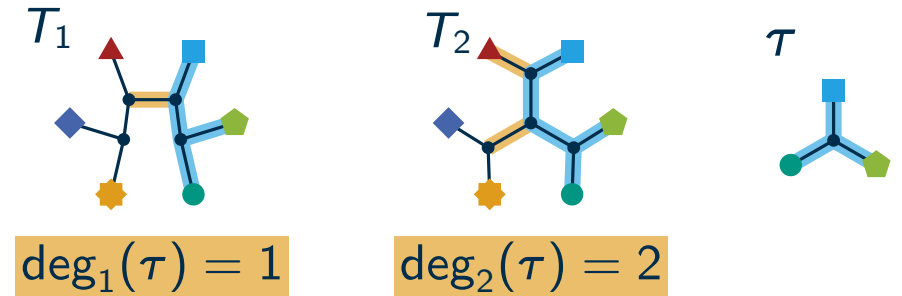
# Rough plan of action

## The degree of a subtree

- contract a tree  $\tau$  of  $F$  in  $T_i$  to a single node ( $i \in \{1, 2\}$ )
- let  $\deg_i(\tau)$  be the degree of the resulting node

## Amortized perspective (as before, but more formal)

- $\tau$  splits  $T_i$  into  $\deg_i(\tau)$  subtrees
- it follows: big degree  $\deg_i(\tau) \Rightarrow$  many trees in  $F$



### Lemma

If  $F$  consists of  $k$  trees, then  $\sum_{\tau \in F} \deg_i(\tau) \leq 2k - 2$ .

(few trees  $\Rightarrow$  small degrees)

### Lemma

Every tree  $\tau \in F$  has at most  $c(\deg_1(\tau) + \deg_2(\tau))$  leaves (after applying rules 1 and 2).

(small degrees  $\Rightarrow$  few leaves)

# The sum of degrees is low

## Lemma

If  $F$  consists of  $k$  trees, then  $\sum_{\tau \in F} \deg_i(\tau) \leq 2k - 2$ .

(few trees  $\Rightarrow$  small degrees)

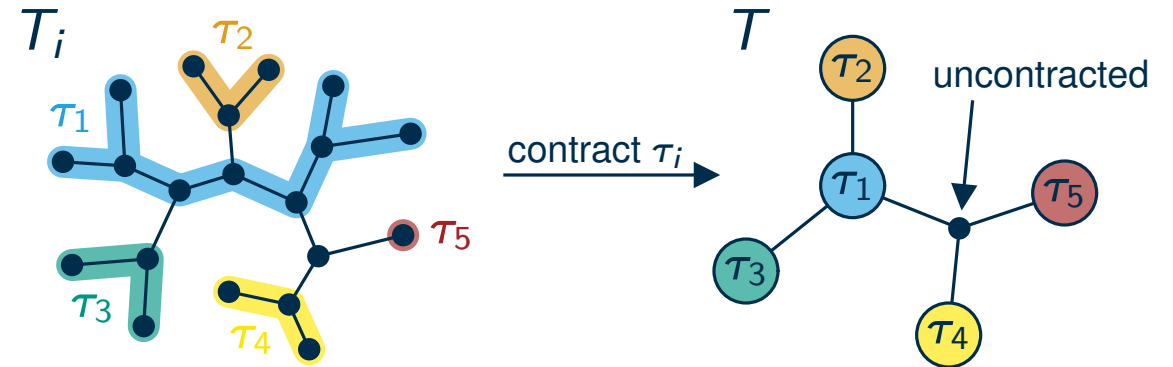
## Proof

- contract each  $\tau$  of  $F$  in  $T_i \rightarrow$  tree  $T$
- summing all degrees in  $T$ :

$$\sum_{v \in V(T)} \deg(v) = 2m = 2n - 2$$

- vertex  $v$  of  $T$  is either: one of the  $k$  trees  $\tau_i$  or an uncontracted vertex of  $T_i$  with degree 3
- let  $n_3$  be the number of uncontracted vertices; then:

$$3n_3 + \sum_{\tau \in F} \deg_i(\tau) = 2(k + n_3) - 2 \quad \Rightarrow \quad n_3 + \sum_{\tau \in F} \deg_i(\tau) = 2k - 2 \quad \Rightarrow \quad \sum_{\tau \in F} \deg_i(\tau) \leq 2k - 2$$



# Trees of low degree have few leaves

## Lemma

(small degrees  $\Rightarrow$  few leaves)

Every tree  $\tau \in F$  has at most  $c(\deg_1(\tau) + \deg_2(\tau))$  leaves (after applying rules 1 and 2).

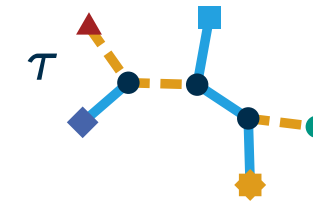
## Two types of edges in $\tau$

- consider how  $\tau$  lies in  $T_1$  and  $T_2$
- each edge of  $\tau$  is an **edge** or a **path** in  $T_i$
- mark it as **path** if it is a path in  $T_1$  or  $T_2$  and **edge** otherwise



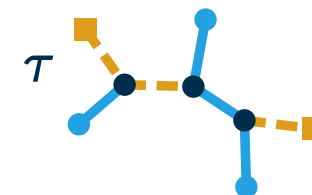
## Why is the labelling useful?

- **#path edges**  $\leq \deg_1(\tau) + \deg_2(\tau)$
- it remains to show: **#leaves**  $\leq c \cdot$  **#path edges**



## Two types of leaves in $\tau$

- classify each leaf by its incident edge: **path leaf** or **edge leaf**



# How do the reduction rules help?

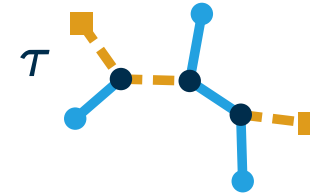
## Lemma

Every tree  $\tau \in F$  has at most  $c(\deg_1(\tau) + \deg_2(\tau))$  leaves (after applying rules 1 and 2).

(small degrees  $\Rightarrow$  few leaves)

## Why is the labelling useful?

- #path edges  $\leq \deg_1(\tau) + \deg_2(\tau)$
- it remains to show: #leaves  $\leq c \cdot$  #path edges

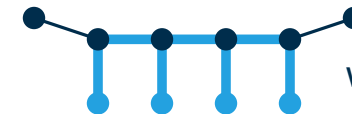


## What do the reduction rules tell us about $\tau$ ?

- rule 1: no cherry with two edge leaves
- rule 2: no “leave-path” of length 4 with only edge edges



would be reduced by rule 1



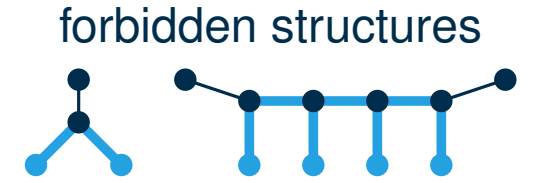
would be reduced by rule 2

## Lemma

Let  $\tau$  be a full binary tree with  $m_r$  orange edges (non-orange edges are blue). If  $\tau$  does not contain one of the two forbidden structures, then it has at most  $10m_r$  leaves.

(implies the above lemma)

# Here comes the proof



## Lemma

(implies: small degrees  $\Rightarrow$  few leaves)

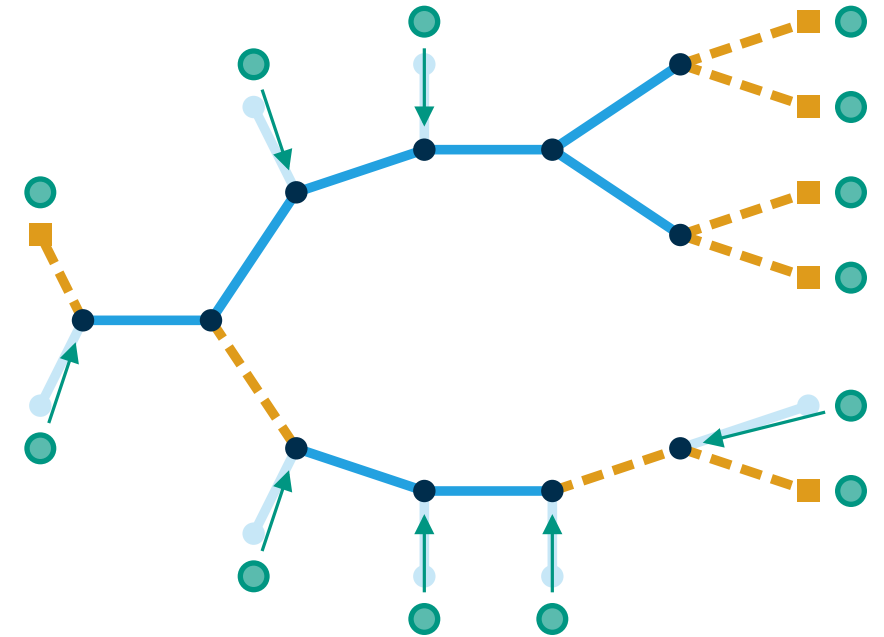
Let  $\tau$  be a full binary tree with  $m_r$  orange edges (non-orange edges are blue). If  $\tau$  does not contain one of the two forbidden structures, then it has at most  $10m_r$  leaves.

## Proof

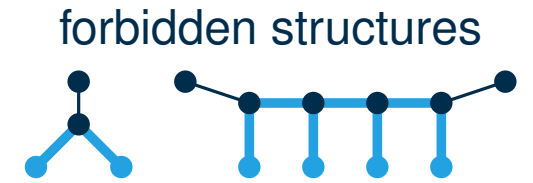
- start with one token  $\bullet$  per leaf  $\rightarrow$  count  $\bullet$
- delete blue leaves and move  $\bullet$  to neighbor
  - only orange leaves and 1  $\bullet$  per leaf and deg-2 vertex

this is ok: each orange leaf is attached to an orange edge

we need to deal with this



# Here comes the proof



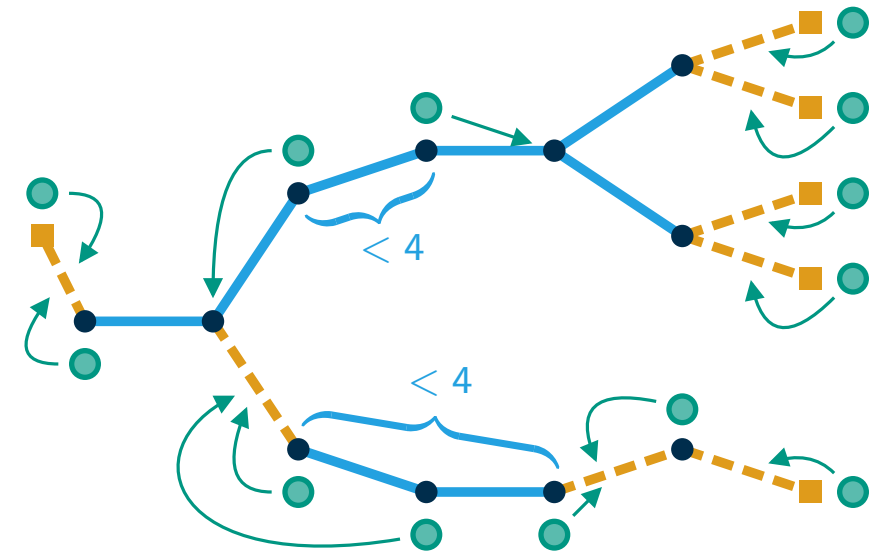
## Lemma

(implies: small degrees  $\Rightarrow$  few leaves)

Let  $\tau$  be a full binary tree with  $m_r$  orange edges (non-orange edges are blue). If  $\tau$  does not contain one of the two forbidden structures, then it has at most  $10m_r$  leaves.

## Proof

- start with one token  $\bullet$  per leaf  $\rightarrow$  count  $\bullet$
- delete blue leaves and move  $\bullet$  to neighbor
  - only orange leaves and 1  $\bullet$  per leaf and deg-2 vertex
  - blue paths of degree-2 vertices have  $< 4$  vertices
- move each  $\bullet$  to closest [deg-3 vertex or orange edge]
  - $\leq 6$   $\bullet$  per deg-3 vertex and  $\leq 4$   $\bullet$  per orange edge
- $\#(\text{deg-3 vertices}) < \#(\text{leaves}) = \#(\text{orange leaves}) \leq m_r$
- $\#(\text{leaves of } \tau) = \#\bullet \leq 6m_r + 4m_r$



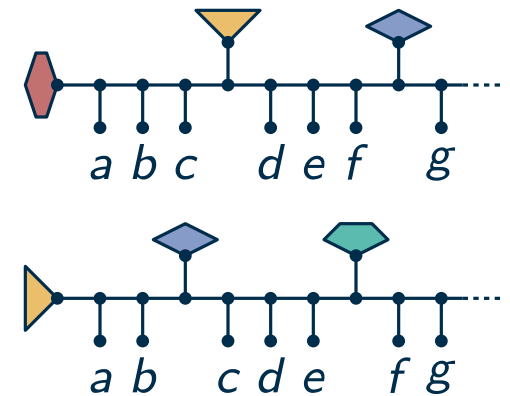
# What have we done?

## Lemma

Exhaustively applying reduction rules 1 and 2 to a yes-instance yields an equivalent instance with at most  $ck$  leaves (for a small constant  $c$ ).

## Proof

- just seen: number of leaves of  $\tau \in F$  is at most  $10(\deg_1(\tau) + \deg_2(\tau))$ 
  - $\tau$  has many leaves  $\Rightarrow$  it splits  $T_1$  or  $T_2$  into many subtrees
- seen before:  $\sum_{\tau \in F} \deg_i(\tau) < 2k$ 
  - $\tau \in F$  split  $T_1$  or  $T_2$  into many subtrees  $\Rightarrow F$  has many trees
- it follows:  $F$  has at most  $40k$  leaves
  - many leaves  $\Rightarrow$  many trees



**Reduction rule 3:** more than  $40k$  leaves after applying rule 1 and 2  $\rightarrow$  no-instance of size  $O(1)$

# Wrap-Up

## Problem: MAXIMUM AGREEMENT FOREST

Given  $T_1, T_2$ , and a parameter  $k$ . Is there an agreement forest of at most  $k$  trees?

## Theorem

MAXIMUM AGREEMENT FOREST has a kernel of size  $O(k)$  that can be computed in poly time.

## Technique

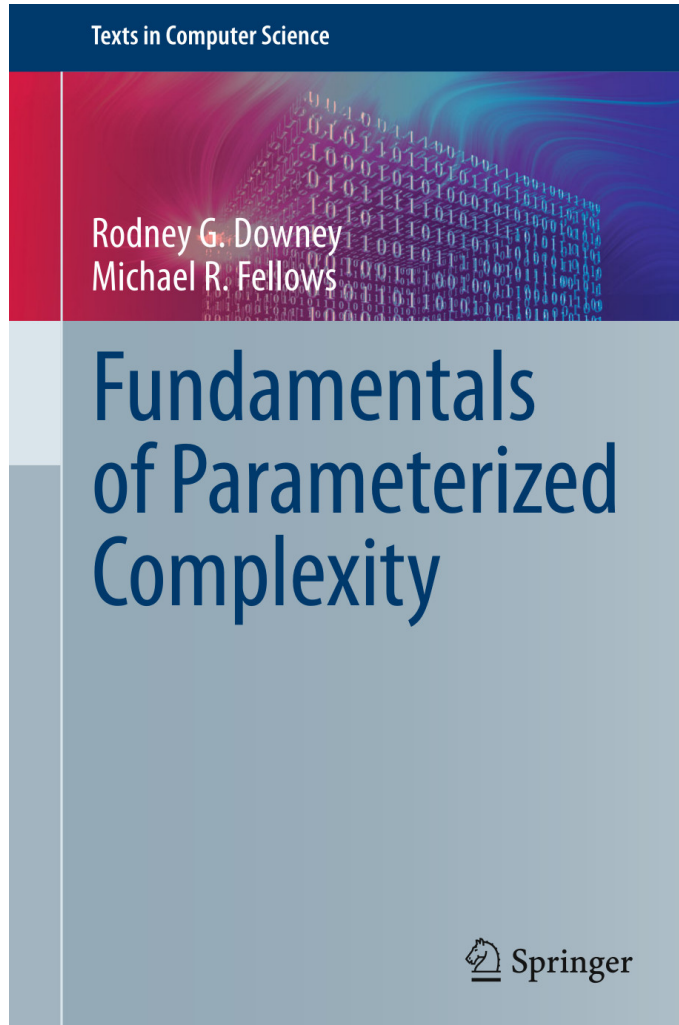
- a specific goal beyond “I want a small kernel” helps (e.g., I want every tree in  $F$  to have few leaves)
- counterexamples tell us how to reduce
- weakening the goal later might be helpful (e.g., amortizing over all trees in  $F$ )

## Not seen today

- specific running time for kernelization
- specific running time for brute force on the kernel

$O(4^k \cdot k^5)$  is possible

# Literature



## Comments

- chapter 4.10 is about the topic from today
- also contains links to the original publication
- freely available for KIT members

[link.springer.com/book/10.1007/978-1-4471-5559-1](https://link.springer.com/book/10.1007/978-1-4471-5559-1)