Local certification meets model checking

Laurent Feuilloley

CNRS & University of Lyon

SIROCCO 2022

June 27th

Local certification meets model checking

Distributed computing

Formal methods

Local certification meets model checking

Distributed computing

- Locality in networks
- Self-stabilization

Formal methods

- automata
- Logic

Local certification meets model checking

Distributed computing

Formal methods

Same goal: Check correctness efficiently

Local certification meets model checking

Distributed computing

Formal methods

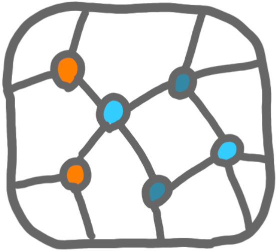
Same goal: Check correctness efficiently

→ Fault-tolerance

→ Critical systems

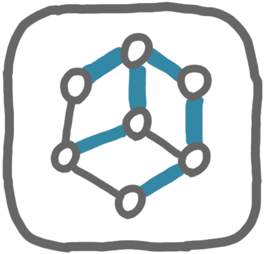
Properties to check

Node labels properties



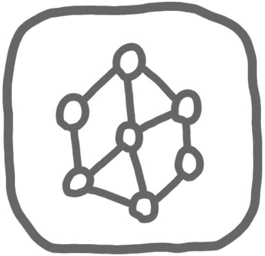
Ex.: k-coloring

Edge labels properties



Ex: Spanning tree

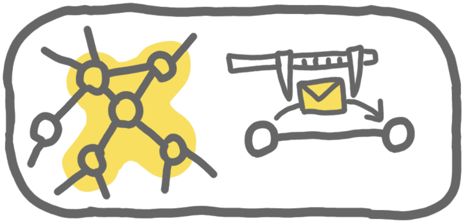
Network properties



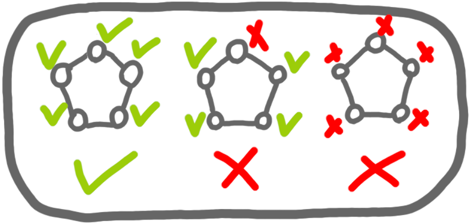
Ex: planarity

Distributed checking

Computational model

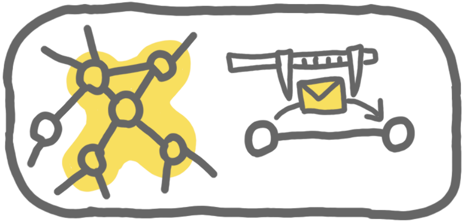


Distributed decision



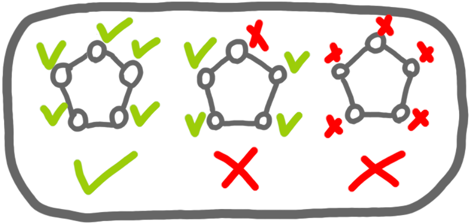
Distributed checking

Computational model



Locality + small messages
(reduce overhead)

Distributed decision



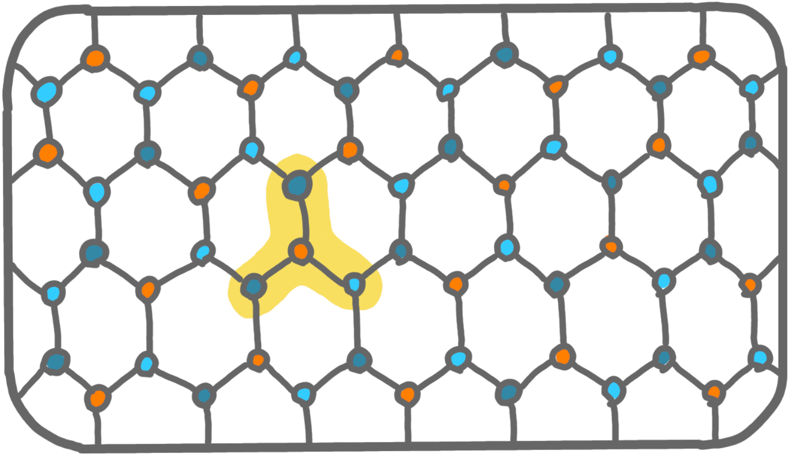
Unanimous accept
(reject → launch reset)

Basic local decision is very limited

k-coloring

Spanning tree

Planarity

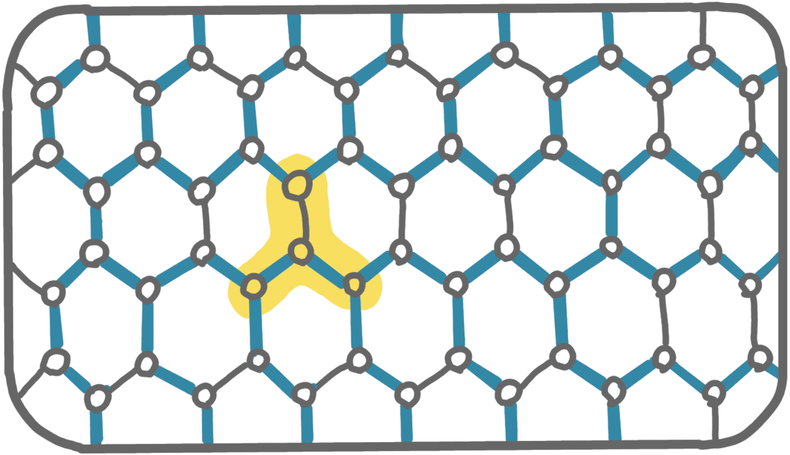


Basic local decision is very limited

k-coloring

Spanning tree

Planarity

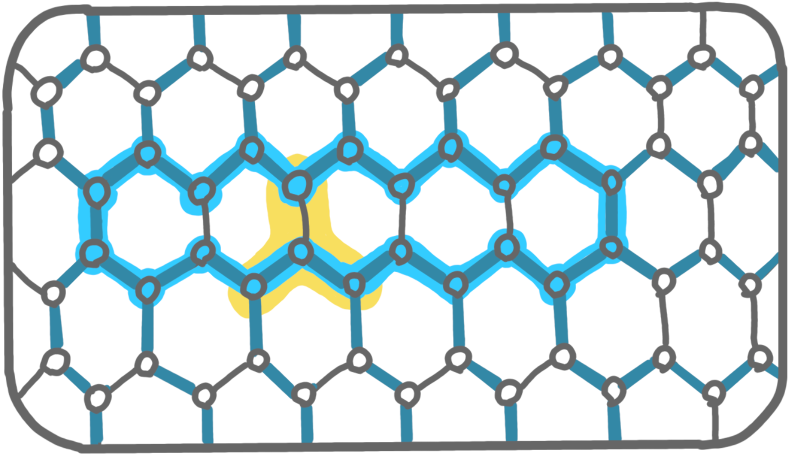


Basic local decision is very limited

k-coloring

Spanning tree

Planarity

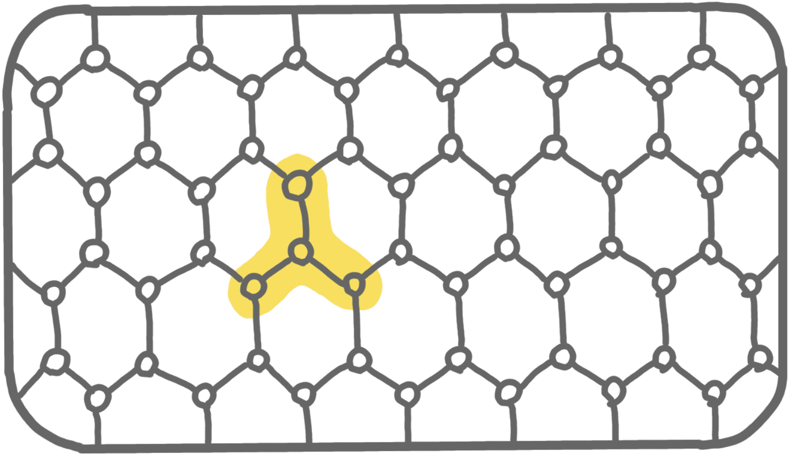


Basic local decision is very limited

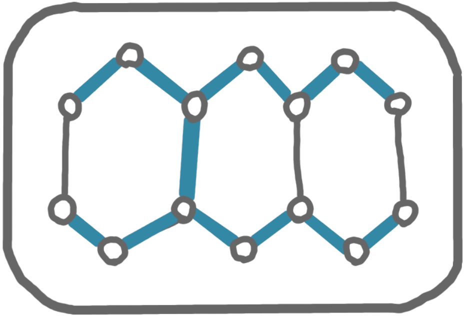
k-coloring

Spanning tree

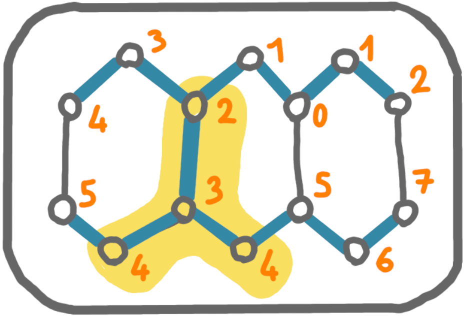
Planarity



A labeling that certifies the property



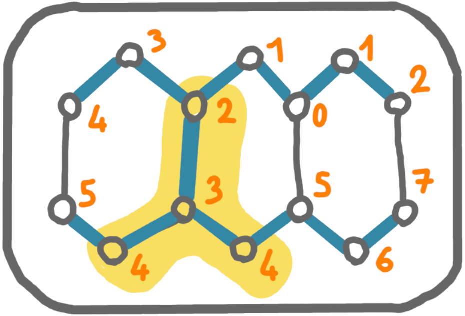
A labeling that certifies the property



Expected labels: distances to a root

Checking: consistency of distances

A labeling that certifies the property



Expected labels:
distances to a root

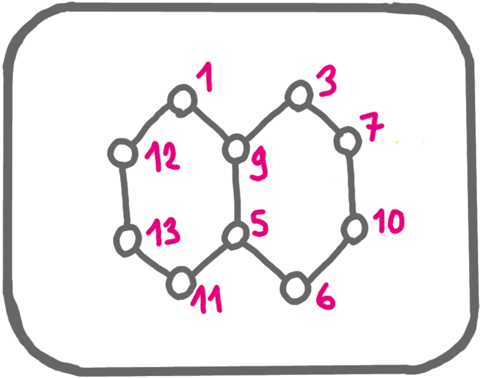
Checking:
consistency of distances

Local certification = A checking procedure s.t. exists accepted labeling $\Leftrightarrow$ the property is satisfied

All properties have a local certification

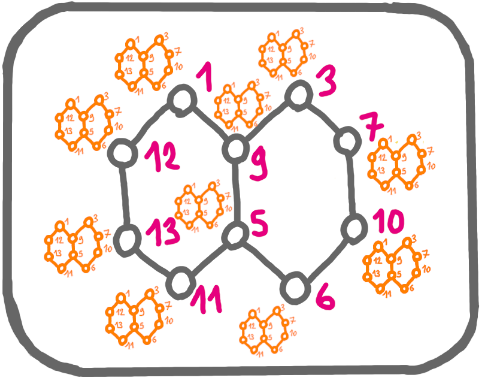
All properties have a local certification

- Nodes equipped with unique IDs



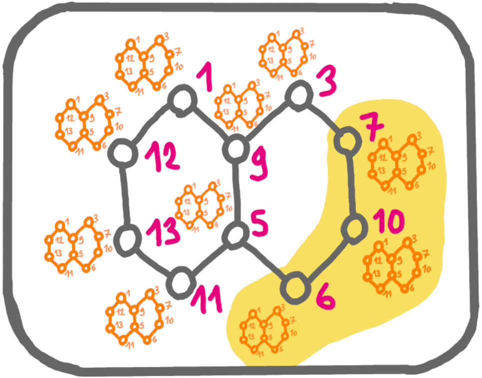
All properties have a local certification

- Nodes equipped with unique IDs
- Labels: map of the graph



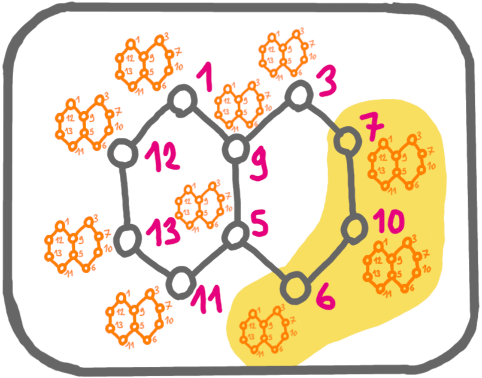
All properties have a local certification

- ▷ Nodes equipped with unique IDs
- ▷ Labels: map of the graph
- ▷ Check: equal + property



All properties have a local certification

- Nodes equipped with unique IDs
- Labels: map of the graph
- Check: equal + property



Problem: huge certificates

Main question

When can we use a compact certification?

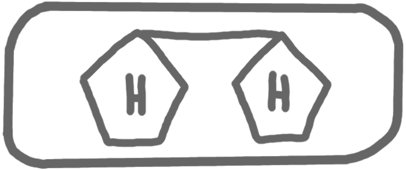
$\equiv$ (poly)log n-size certificates

Main question

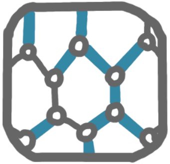
When can we use a compact certification?

≡ (poly)log n-size certificates

Not always ... but often



Symmetry



Sp. tree

Planarity

.....

The model checking approach

Typical theorem:

It is possible to check efficiently that that a logical formula of FORMULA CLASS is satisfied on structures of STRUCTURE CLASS

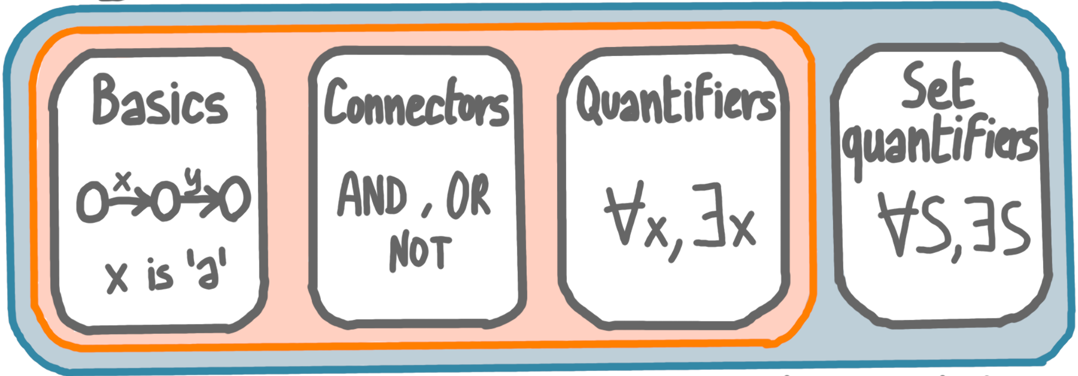
The model checking approach

Typical theorem:

It is possible to check efficiently that
that a logical formula of FORMULA CLASS
is satisfied on structures of STRUCTURE CLASS

Example: FIRST-ORDER (FO) & WORDS

Logic formulas on words



Examples:
- ▷ every 'a' is followed by a 'b' ∈ FO
- ▷ only 'a' and even length ∈ MSO

Theorem and application to certification

Theorem: For properties on words, expressible in MSO = recognizable by an automaton

Theorem and application to certification

Theorem: For properties on words, expressible in MSO = recognizable by an automaton

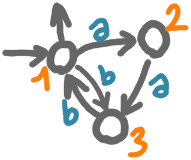
Consequence: MSO properties can be certified with $O(1)$ bits in edge-labeled paths.

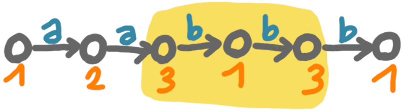
Theorem and application to certification

Theorem: For properties on words, expressible in MSO = recognizable by an automaton

Consequence: MSO properties can be certified with $O(1)$ bits in edge-labeled paths.

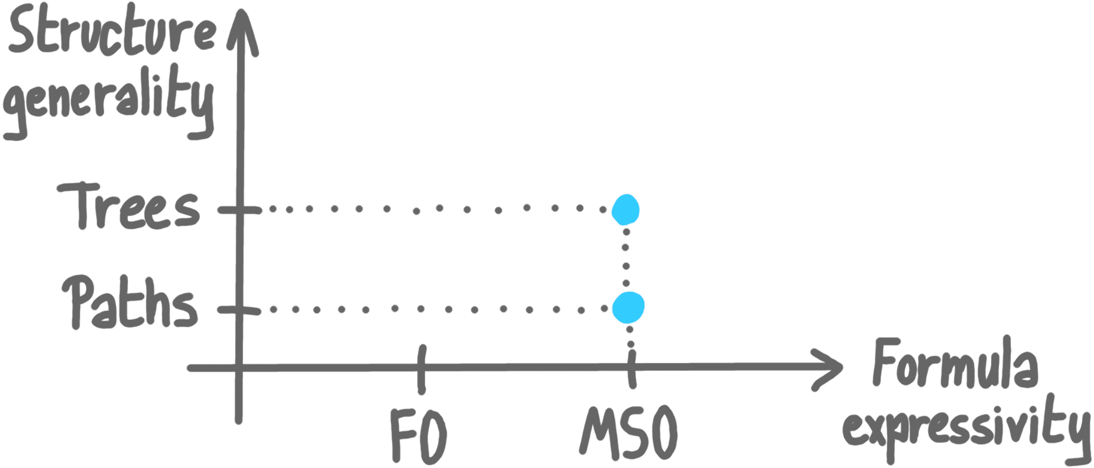
MSO Formula $\xrightarrow{\text{Th.}}$



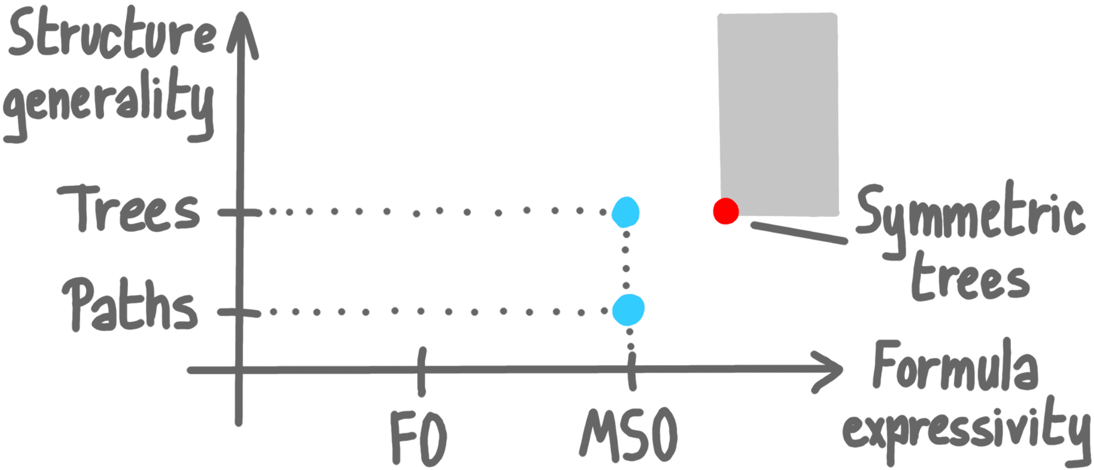


Compact certification beyond MSO + paths?

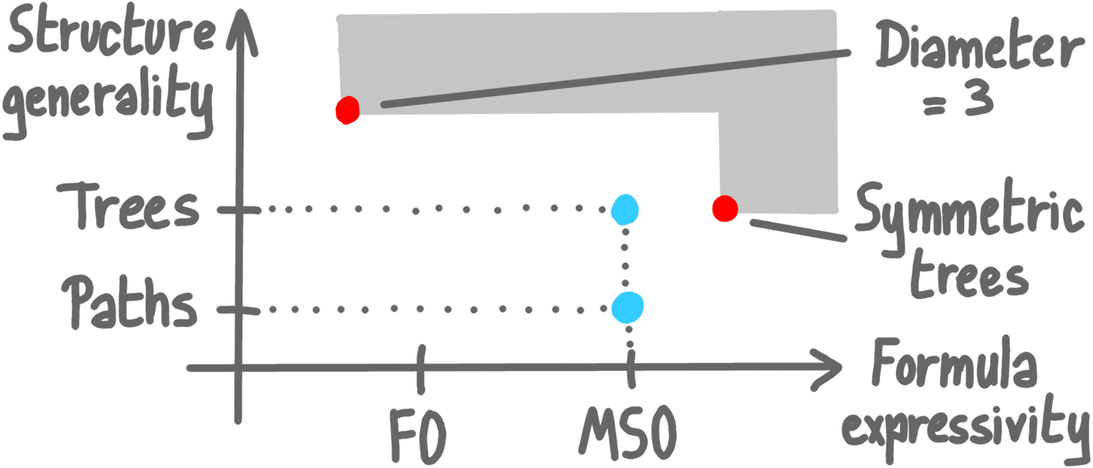
Compact certification beyond MSO+paths?



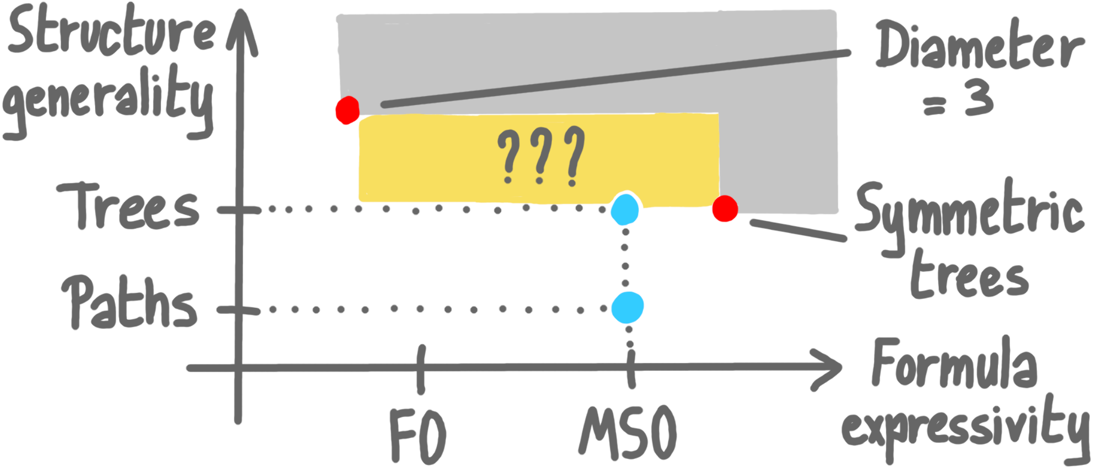
Compact certification beyond MSO+paths?



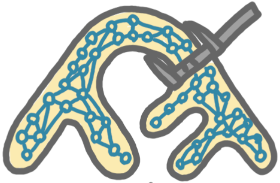
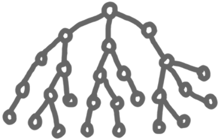
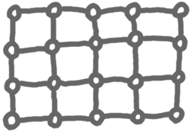
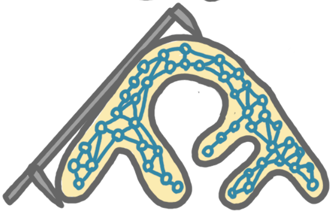
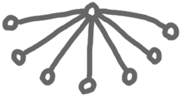
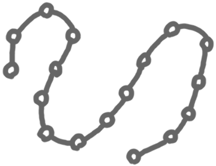
Compact certification beyond MSO+paths?



Compact certification beyond MSO+paths?



Bounding some graph parameters

Parameter	Definition	Examples (small, large)	
Treewidth			
Treedepth			

Theorems (old and new)

Thms (old): MSO model checking takes:

- ▷ IF treewidth $\leq t$, $O(f(t) \cdot n)$ (enormous f)
- ▷ IF treedepth $\leq t$, $O(g(t) \cdot n)$ (reasonable g)

Theorems (old and new)

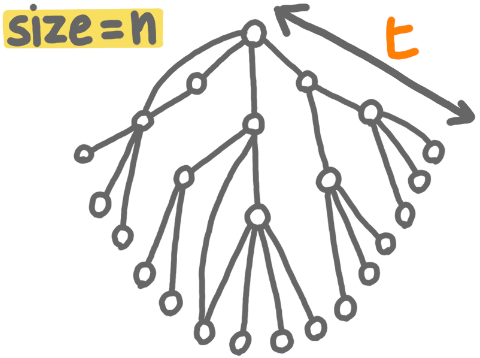
Thms (old): MSO model checking takes:

- ▷ If treewidth $\leq t$, $O(f(t) \cdot n)$. (enormous f)
- ▷ If treedepth $\leq t$, $O(g(t) \cdot n)$. (reasonable g)

Thms (new): MSO certification takes:

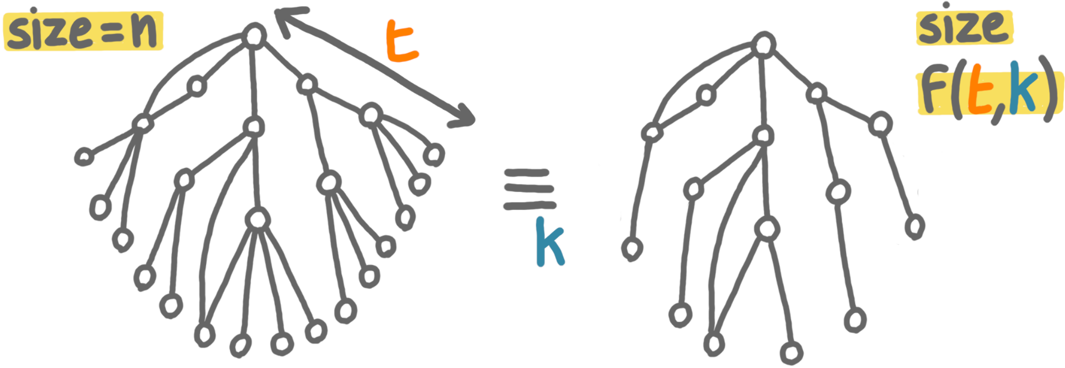
- ▷ If treewidth $\leq t$, $O(\log^2 n)$.
- ▷ If treedepth $\leq t$, $\Theta(\log n)$.

A technique: kernelization



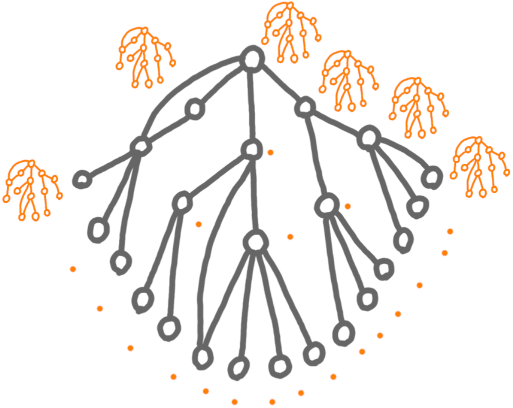
$$\varphi = \exists x_1 \forall x_2 \cdots \exists x_k \cdots$$

A technique: kernelization



$$\varphi = \exists x_1 \forall x_2 \cdots \exists x_k \cdots$$

A technique: kernelization



$\varphi = \exists x_1 \forall x_2 \cdots \exists x_k \cdots$

Local certification of φ

- labels: kernel  + kernel certification
- checking: local consistency + φ is satisfied in the kernel

Application to the forbidden minor question

$\mathcal{H}$ = a finite set of graphs

$\mathcal{C}_{\mathcal{H}} = \{G : \forall H \in \mathcal{H},\ H \text{ is } \underline{\text{not}} \text{ a minor of } G\}$

<u>Question</u>: Is it true that, for all $\mathcal{H}$,
$\mathcal{C}_{\mathcal{H}}$ has a compact certification?

Application to the forbidden minor question

$\mathcal{H}$ = a finite set of graphs

$\mathcal{C}_{\mathcal{H}} = \{G : \forall H \in \mathcal{H}, \ H$ is <u>not</u> a minor of $G\}$

<u>Question:</u> Is it true that, for all $\mathcal{H}$, $\mathcal{C}_{\mathcal{H}}$ has a compact certification?

MSO + treewidth $\rightarrow$ IF $\mathcal{H} \ni$ Planar, yes, $O(\log^2 n)$

MSO + treedepth $\rightarrow$ IF $\mathcal{H} \ni$ Path, yes, $O(\log n)$