# Complexity Landscape for Local Certification

**Nicolas Bousquet**

(Joint work with Laurent Feuilloley, Sébastien Zeitoun)

https://arxiv.org/abs/2505.20915

McGill, June, 2025

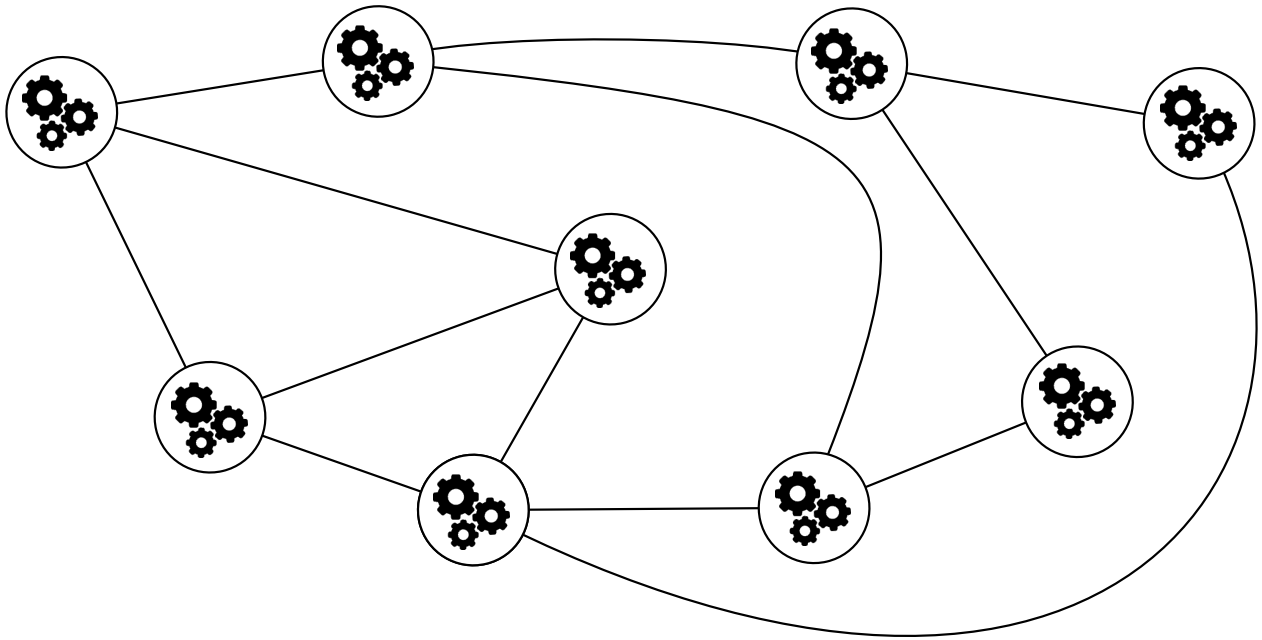
LACIM
Laboratoire d'algèbre, de combinatoire et d'informatique mathématique

# Local certification

# Local certification

**Context :** distributed computing

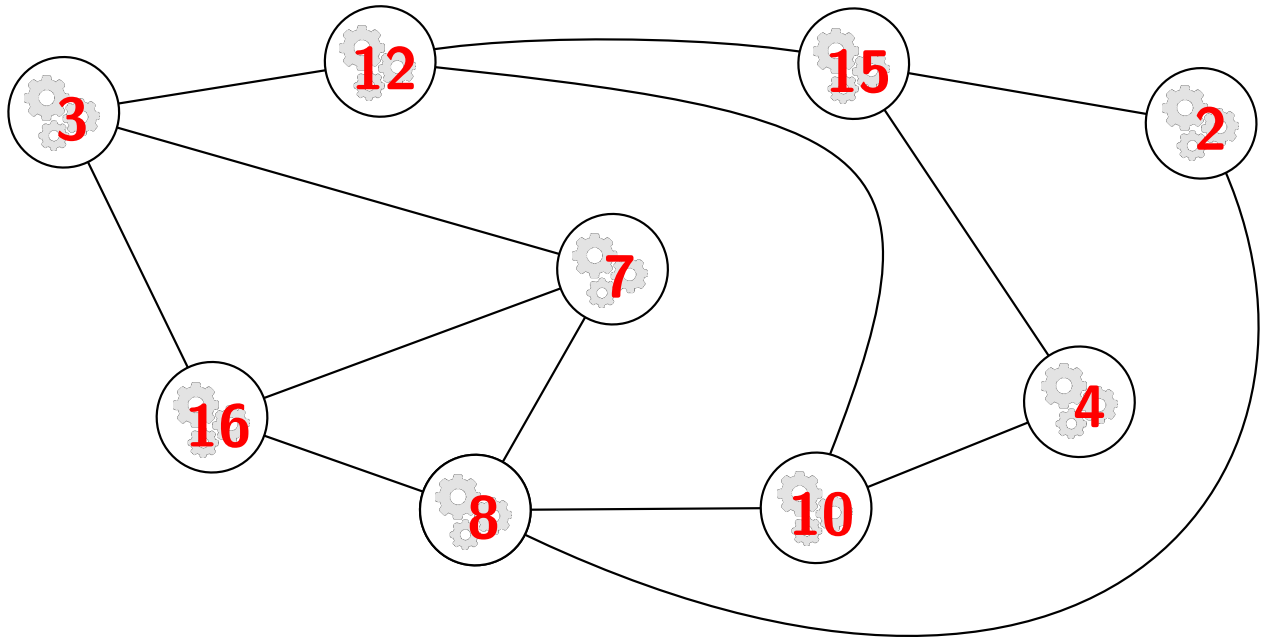
**Model :** graph, $\begin{cases} \text{vertices} = \text{comp. units} \\ \text{edges} = \text{com. channels} \end{cases}$



# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

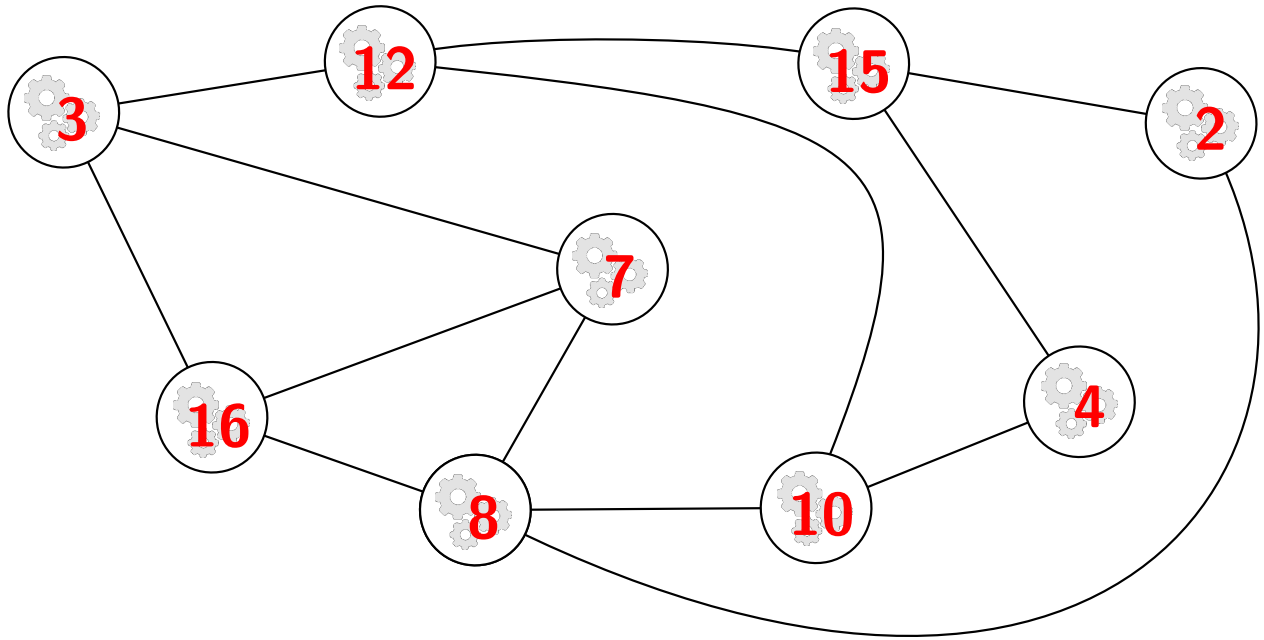


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

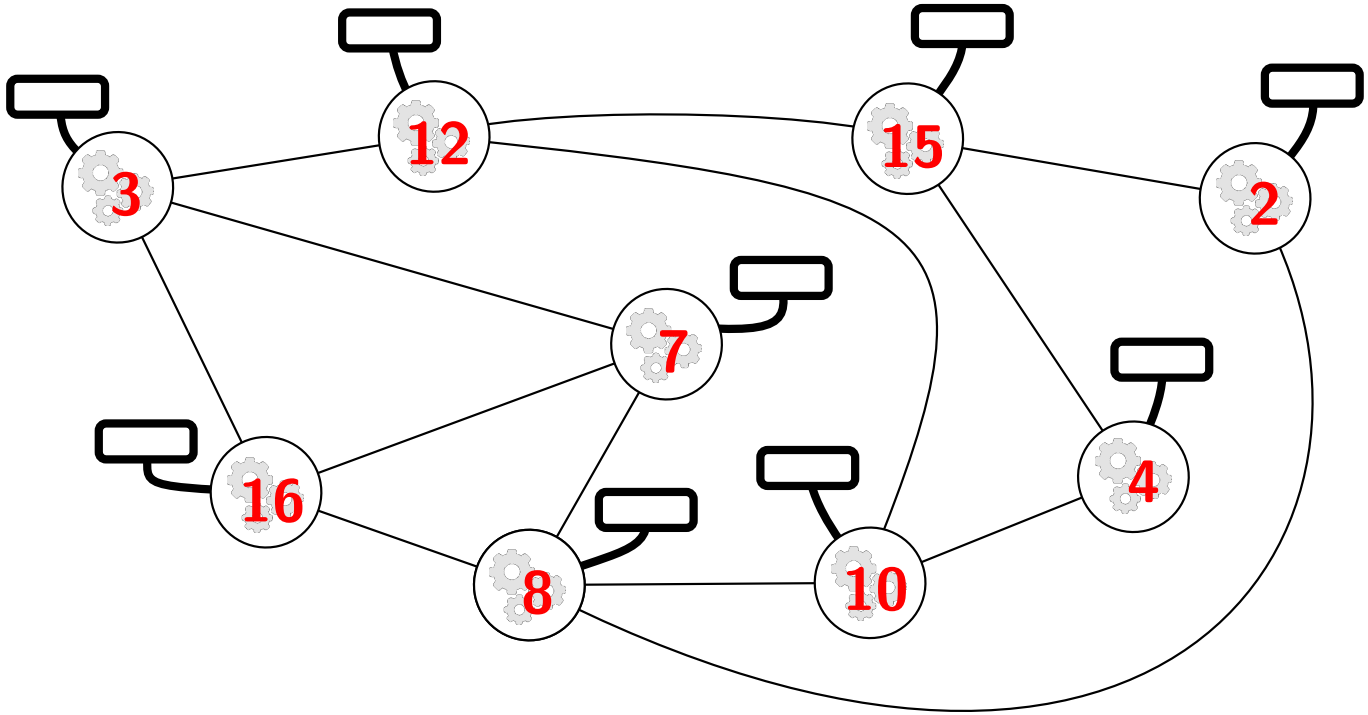


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

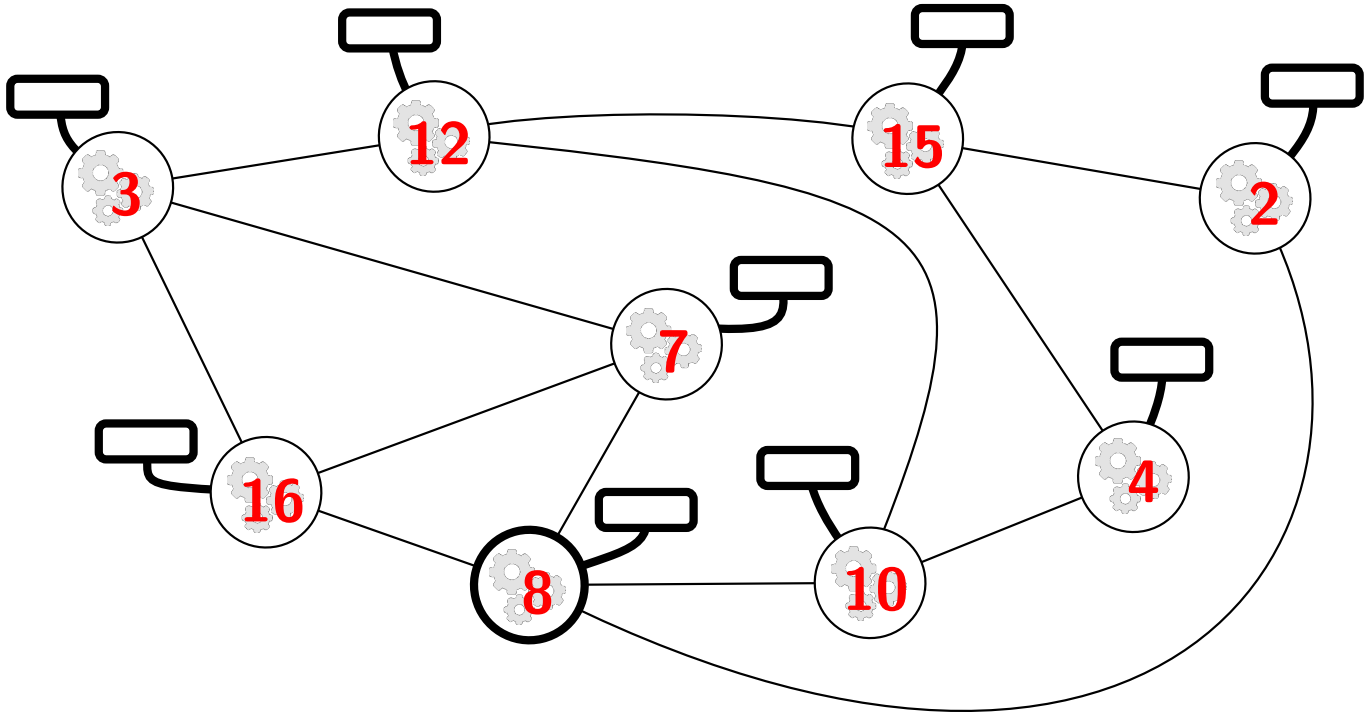


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

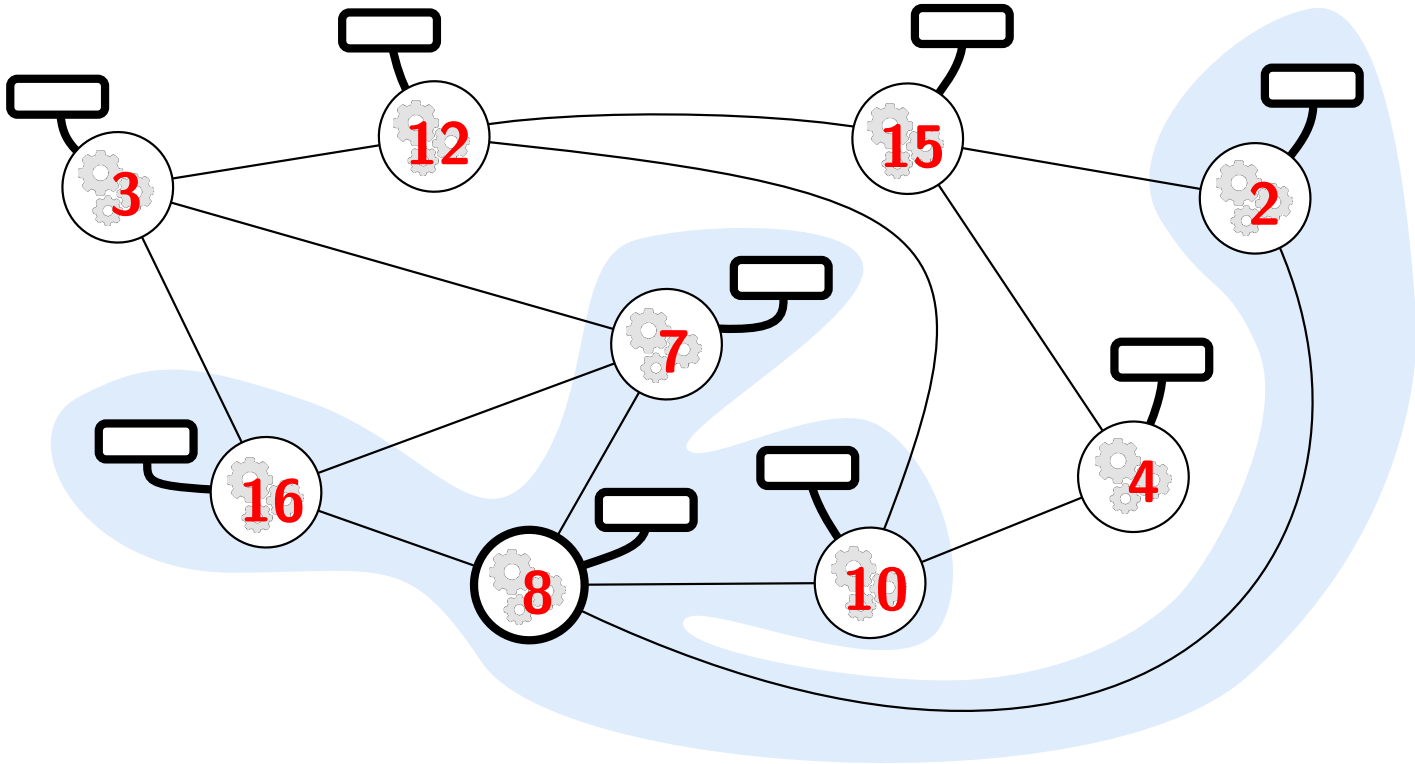


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

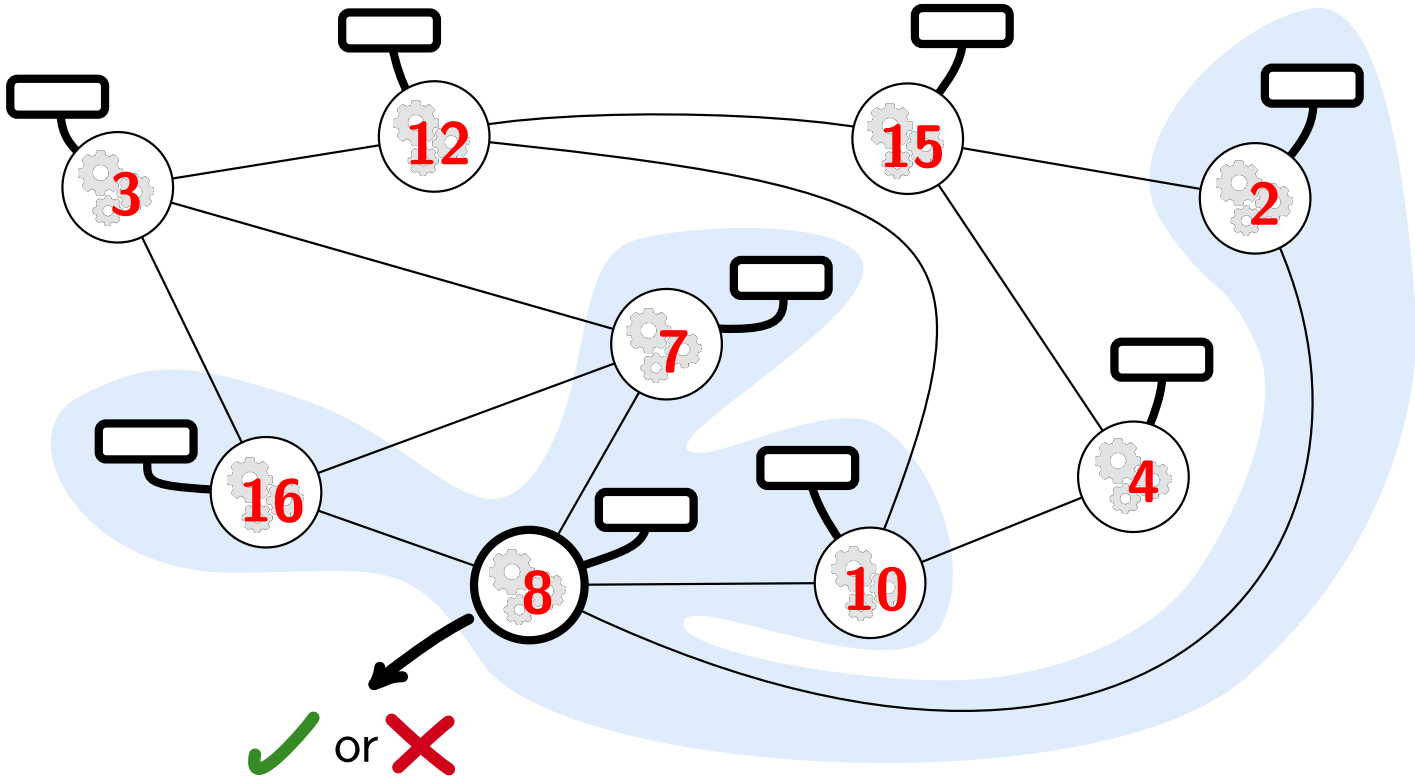


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

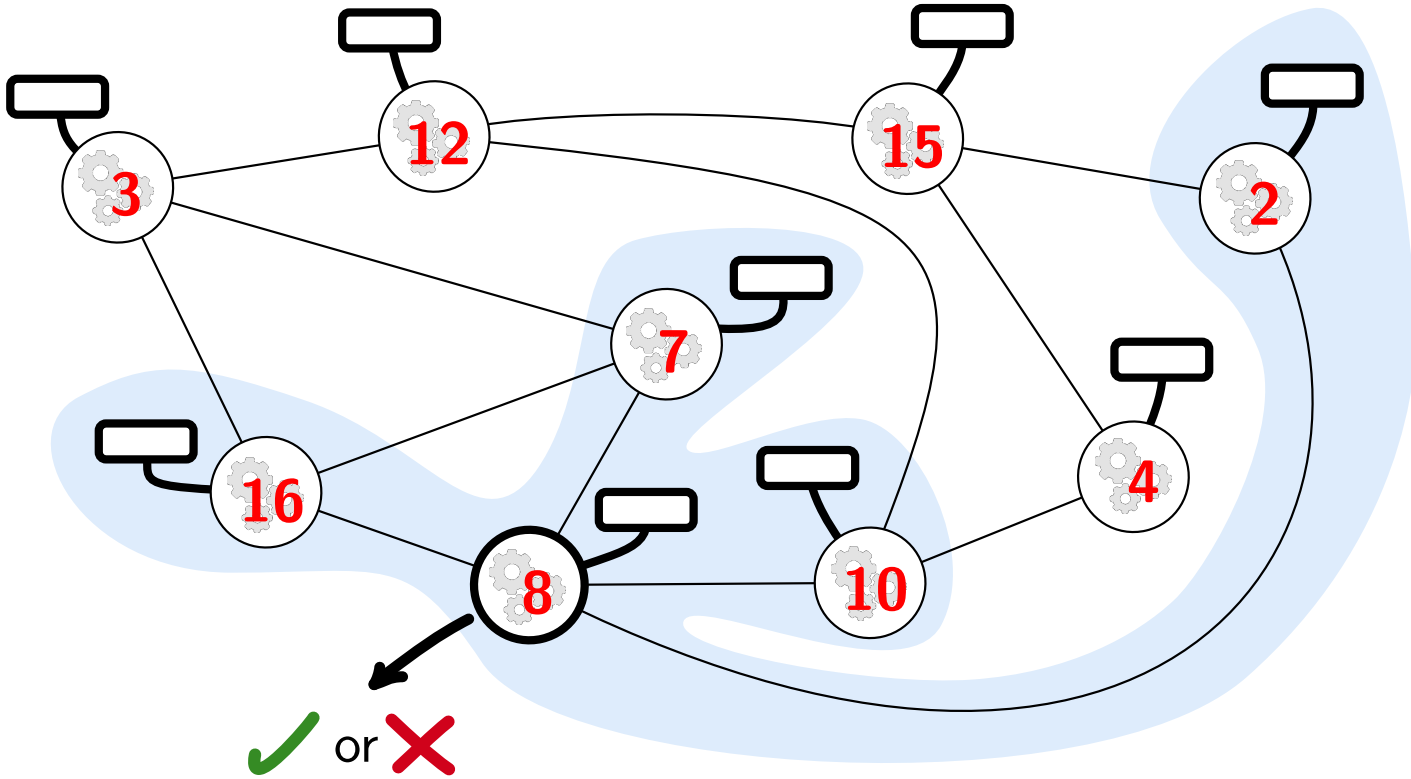


# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates



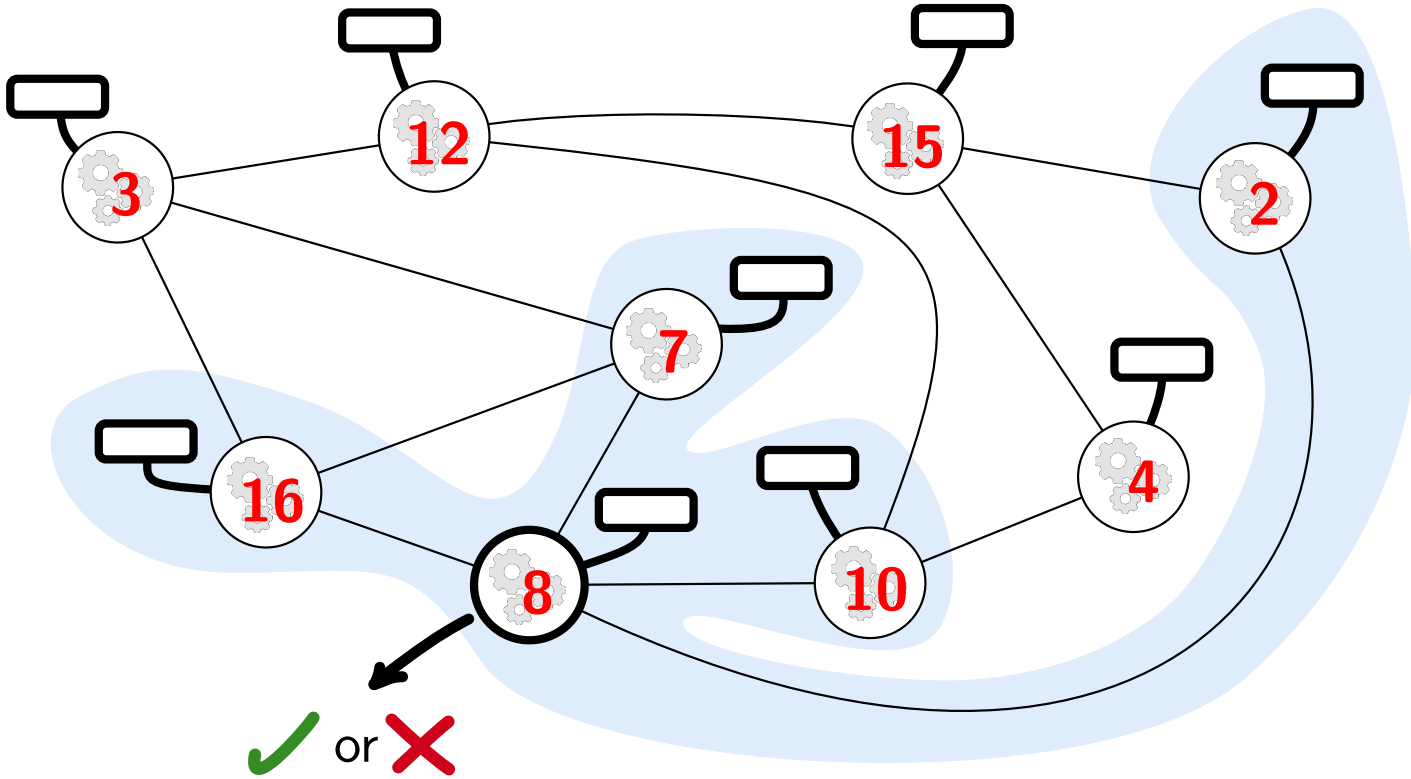

Graph (globally) accepted $\iff$ all the vertices accept (consensus)

# Local certification

**Context :** distributed computing

**Model :**graph, $\begin{cases} \text{vertices} = \text{comp. units with unique ID} \in \{1, \ldots, n^c\} \\ \text{edges} = \text{com. channels} \end{cases}$

**Goal :** verify locally a graph property $\mathcal{P}$, thanks to certificates

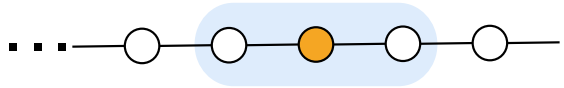



Graph (globally) accepted $\iff$ all the vertices accept (consensus)

$G$ satisfies $\mathcal{P}$ $\iff$ there exists an assignment of the certificates such that $G$ is accepted

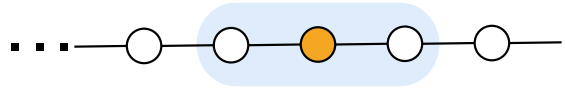
# Certifying paths

# Certifying paths



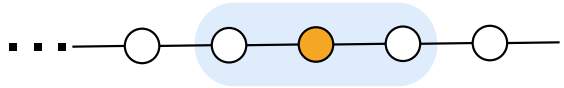
$\longrightarrow$ Path ? Cycle ?

# Certifying paths

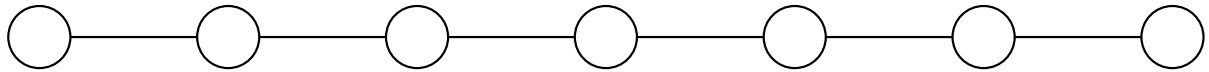



Certificate : distance to a fixed endpoint.

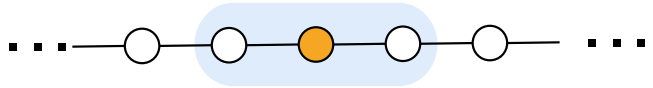
# Certifying paths



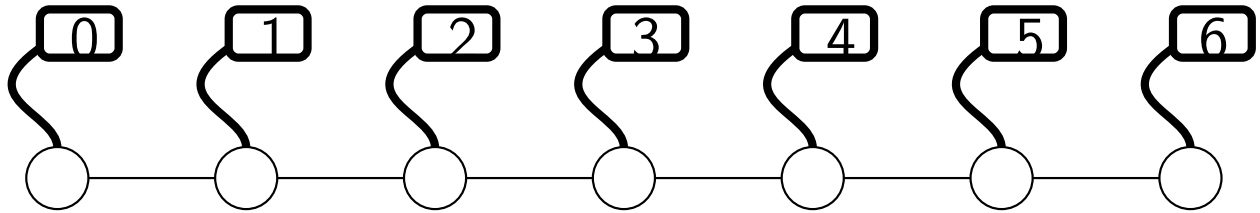

Certificate : distance to a fixed endpoint.



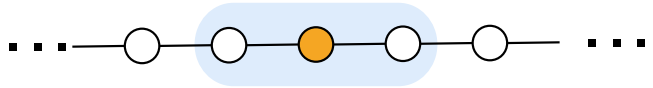
# Certifying paths



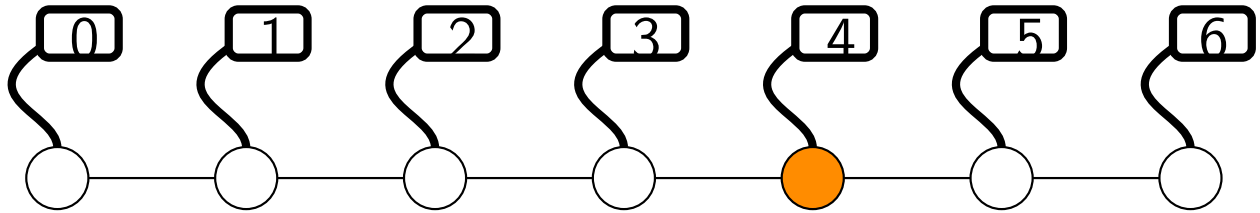

Certificate : distance to a fixed endpoint.



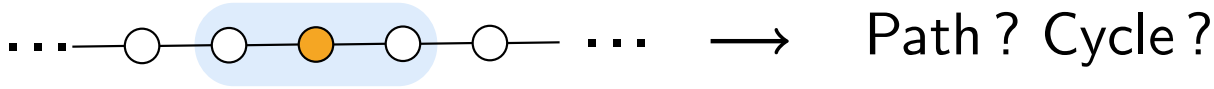
# Certifying paths



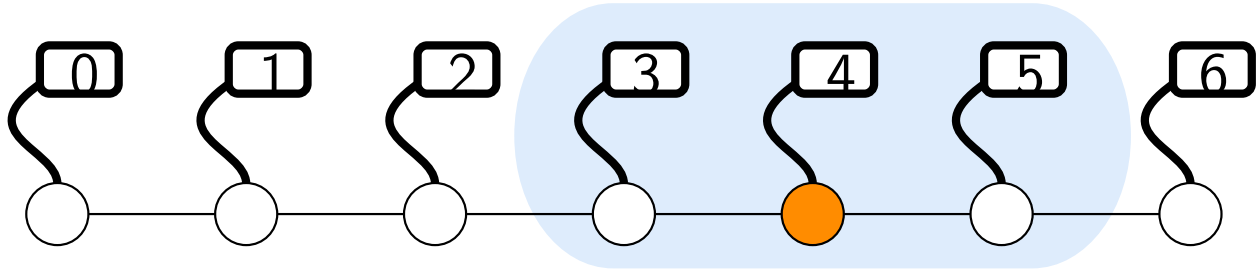

Certificate : distance to a fixed endpoint.



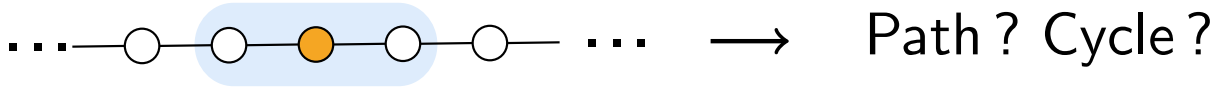
# Certifying paths



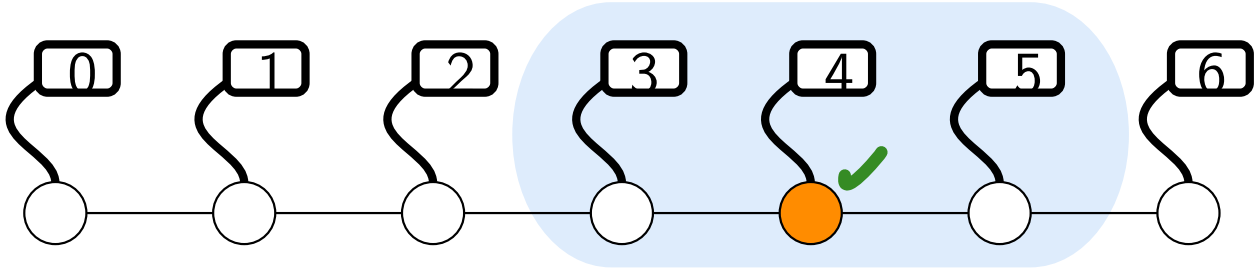

Certificate : distance to a fixed endpoint.



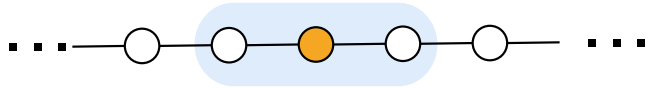
# Certifying paths




Certificate : distance to a fixed endpoint.

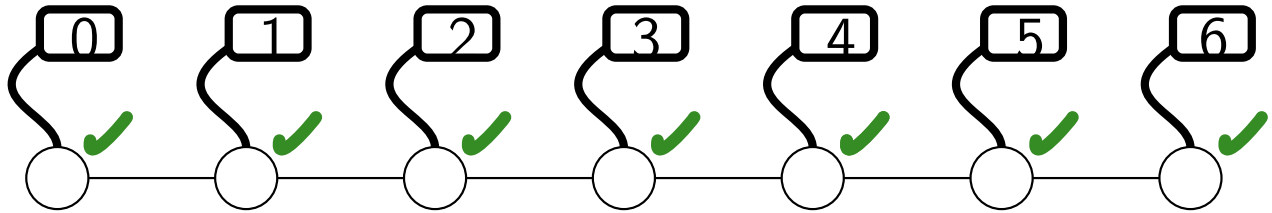


# Certifying paths

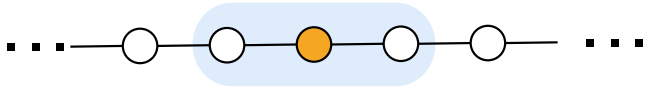


$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.

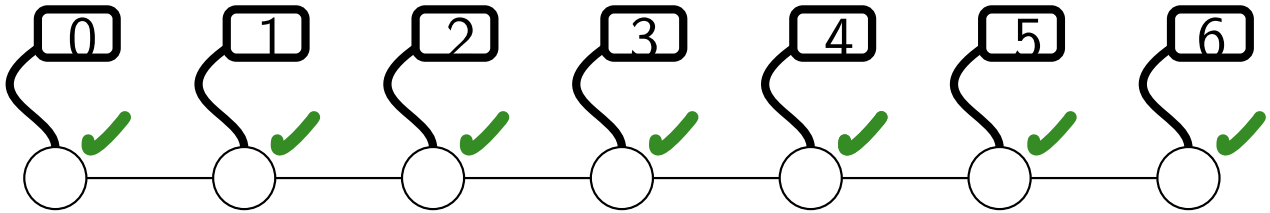


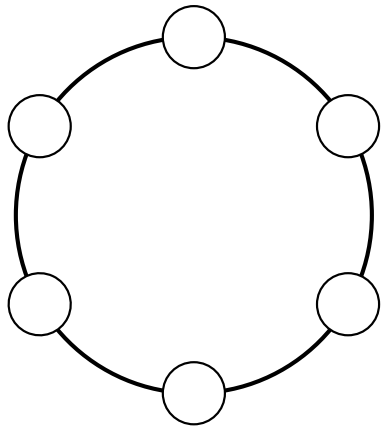
# Certifying paths



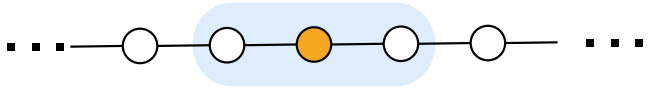
$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.



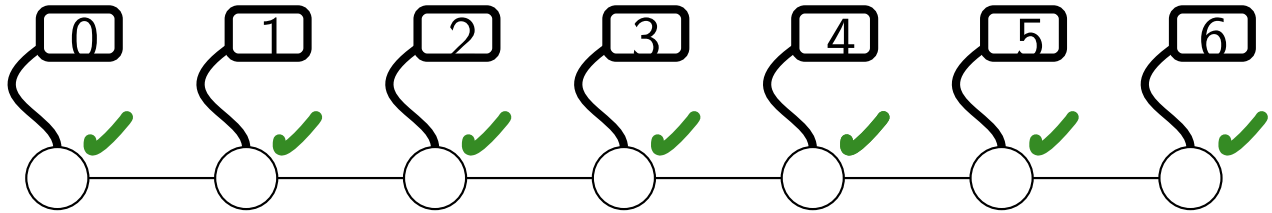



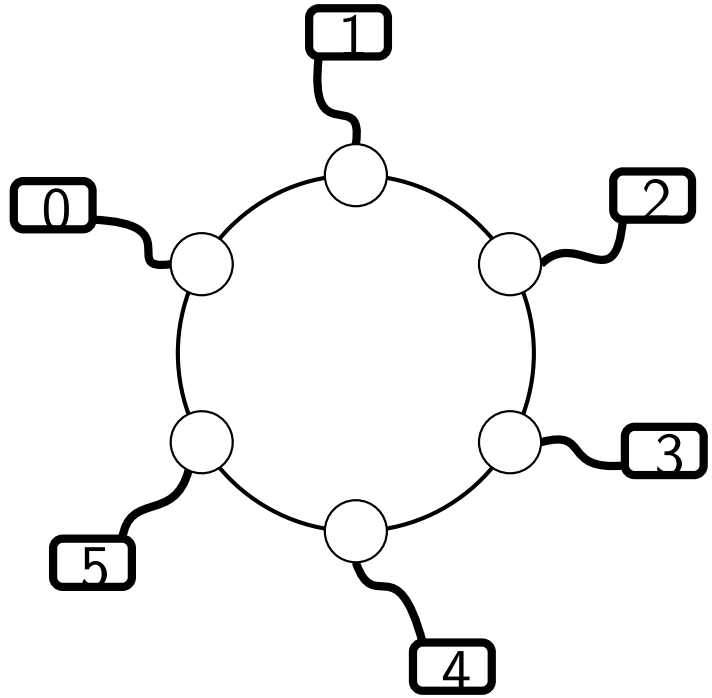
# Certifying paths



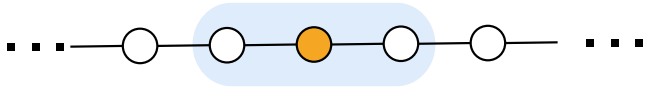
$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.



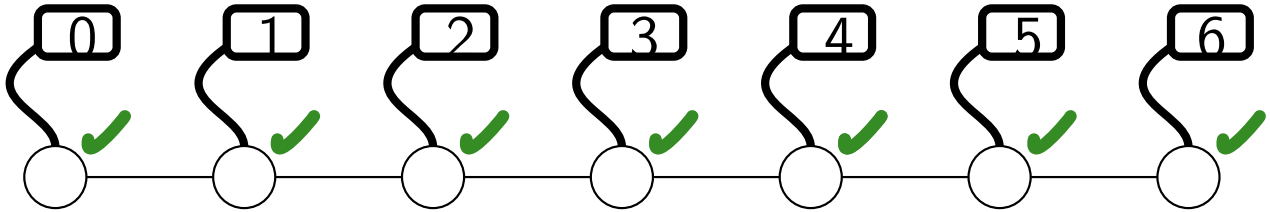



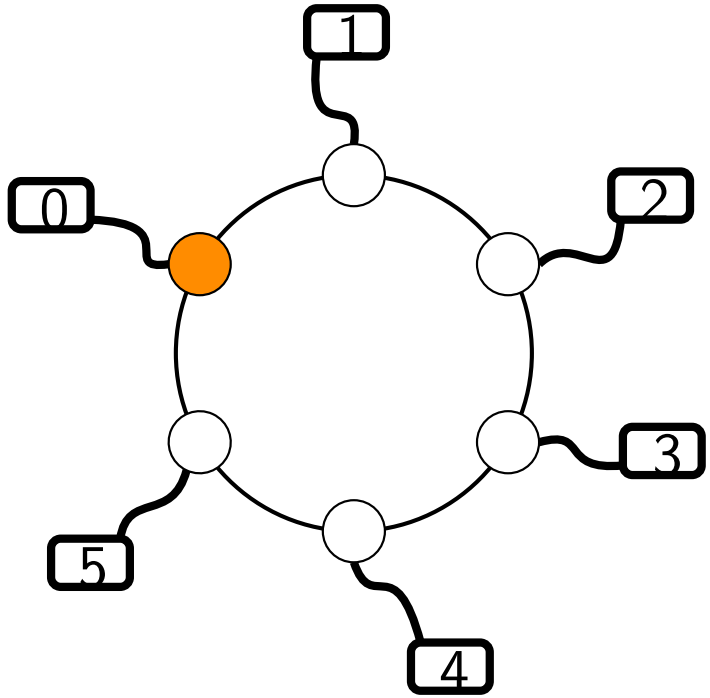
# Certifying paths



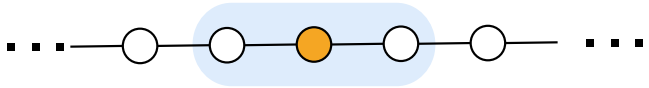
$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.



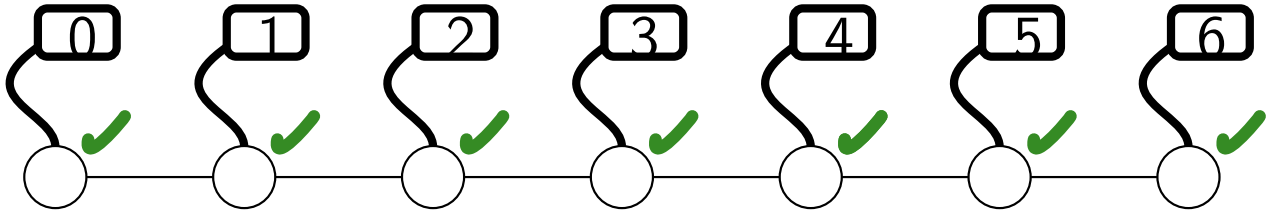



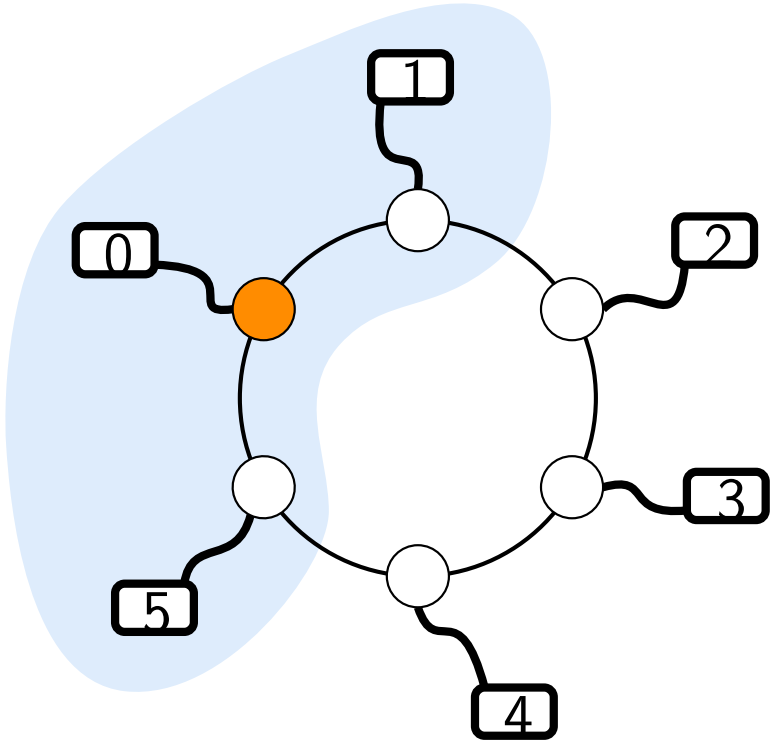
# Certifying paths



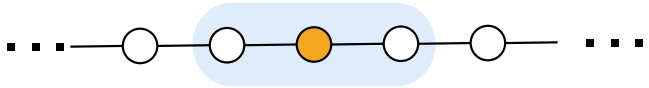
$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.



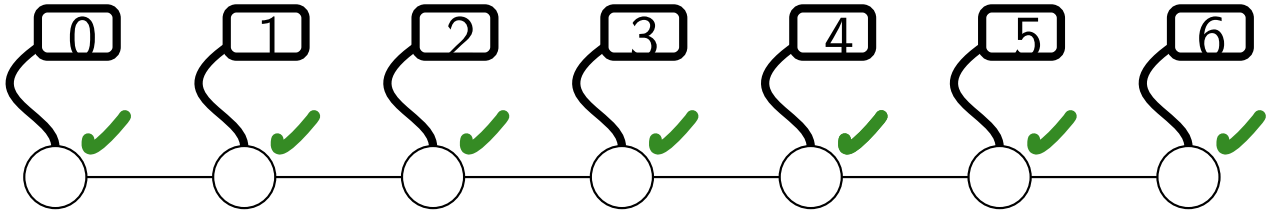



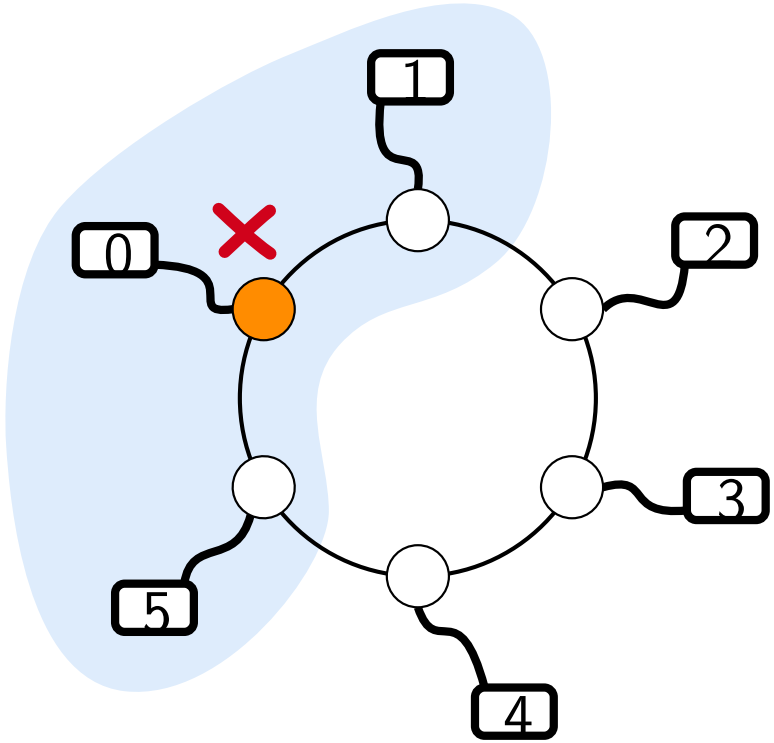
# Certifying paths



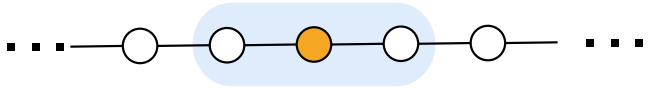
$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.

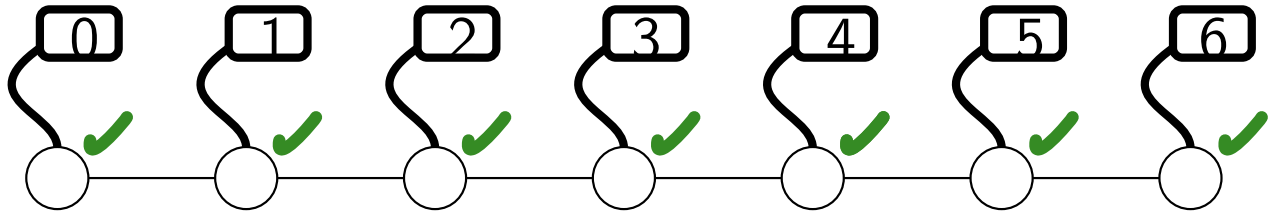


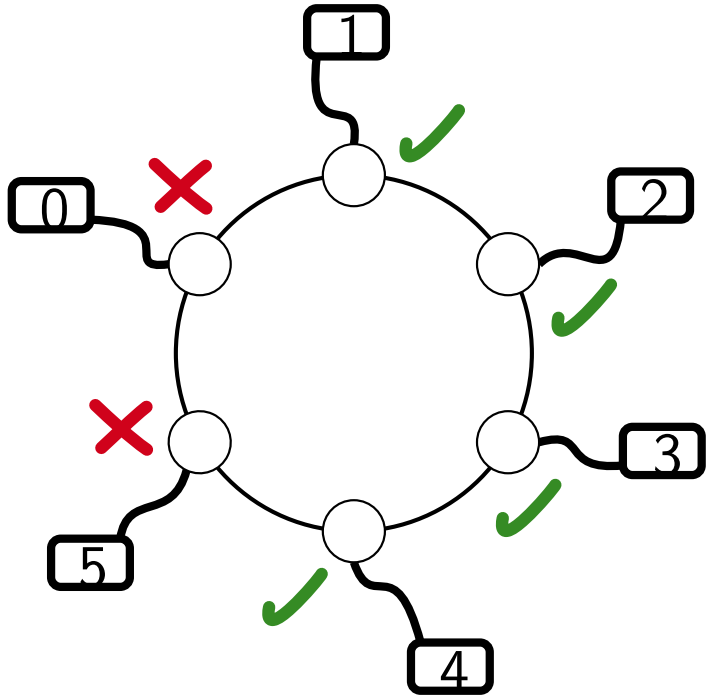



# Certifying paths

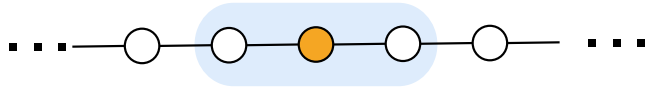



Certificate : distance to a fixed endpoint.



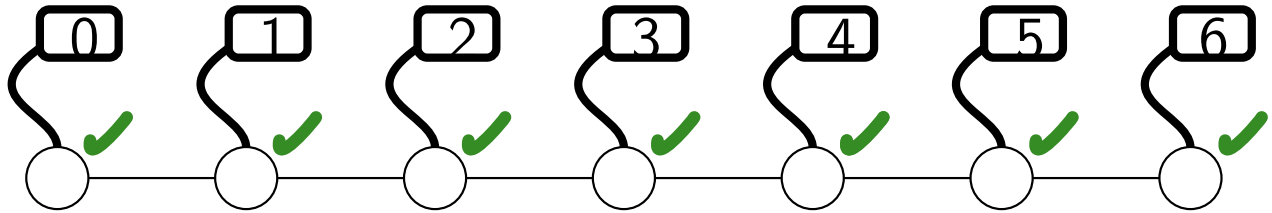



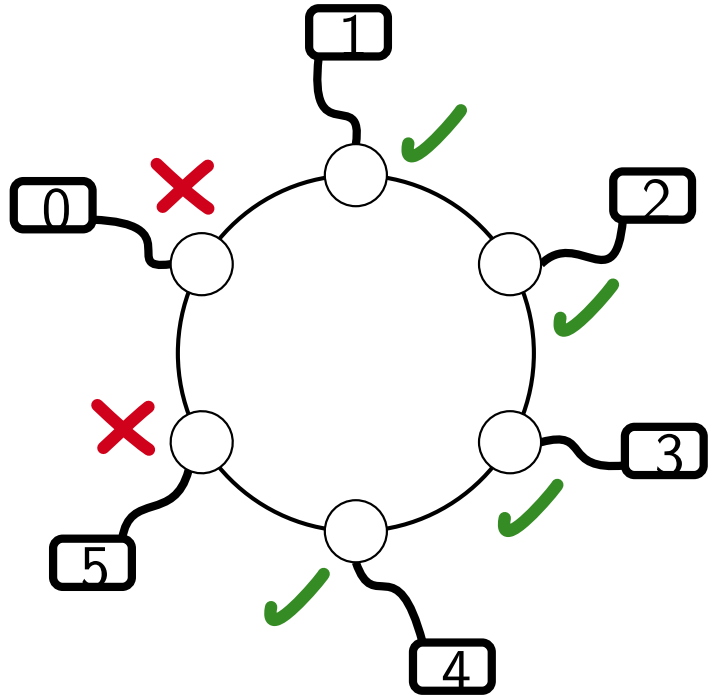
# Certifying paths



$\longrightarrow$ Path ? Cycle ?

Certificate : distance to a fixed endpoint.




Size of the certificates : $\lceil \log n \rceil$

# Certificate size

**Lemma** :
Any property can be certified with $O(n^2)$ bits.

# Certificate size

**Lemma** :
Any property can be certified with $O(n^2)$ bits.

| $\theta(\log n)$ | $\theta^*(n)$ | $\Omega^*(n^2)$ |
|---|---|---|
| Planar | Diameter 3 | Non-trivial isomorphism |
| Bounded genus | | Non 3-colorability |
| Bd treewidth | | |

# Certificate size

**Lemma** :
Any property can be certified with $O(n^2)$ bits.

| $\theta(\log n)$ | $\theta^*(n)$ | $\Omega^*(n^2)$ |
|---|---|---|
| Planar | Diameter 3 | Non-trivial isomorphism |
| Bounded genus | | Non 3-colorability |
| Bd treewidth | | |

**Question :** Which "space complexity" can exist ?

# Certificate size

**Lemma** :
Any property can be certified with $O(n^2)$ bits.

| $\theta(\log n)$ | $\theta^*(n)$ | $\Omega^*(n^2)$ |
|---|---|---|
| Planar | Diameter 3 | Non-trivial isomorphism |
| Bounded genus | | Non 3-colorability |
| Bd treewidth | | |

**Question :** Which "space complexity" can exist ?

**Partial answer :** Any complexity between $\theta(\log n)$ and $\theta^*(n^2)$

# Certificate size

**Lemma** :
Any property can be certified with $O(n^2)$ bits.

| $\theta(\log n)$ | $\theta^*(n)$ | $\Omega^*(n^2)$ |
|---|---|---|
| Planar | Diameter 3 | Non-trivial isomorphism |
| Bounded genus | | Non 3-colorability |
| Bd treewidth | | |

**Question :** Which "space complexity" can exist ?

**Partial answer :** Any complexity between $\theta(\log n)$ and $\theta^*(n^2)$

**Question :** What happens below $\log n$ ?

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

**No gap :**

- For caterpillars

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

**No gap :**

- For caterpillars
- For trees (for $d$)

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

**No gap :**

- For caterpillars
- For trees (for $d$)
- For caterpillars (in $d$) for verification radius 2

# Our main results

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

- Tight : Some properties need $\Omega(\log \log n)$ bits !
- Extension to trees (where $n \leftarrow$ diameter $d$)
- Holds for larger verification radius

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

**No gap :**

- For caterpillars
- For trees (for $d$)
- For caterpillars (in $d$) for verification radius 2
- For labeled paths

# Paths & Automata

**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

# Paths & Automata

**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**

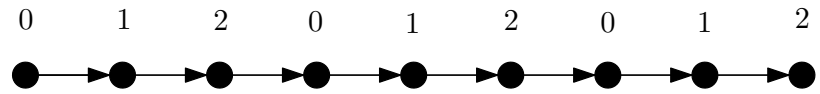
- Certify an orientation of the path.

# Paths & Automata

**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**

- Certify an orientation of the path.



# Paths & Automata

**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**

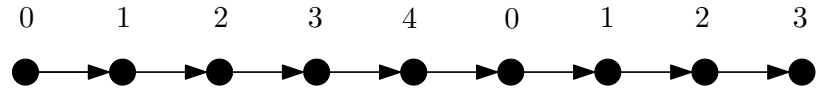
- Certify an orientation of the path.
- Check coherence locally.

# Paths & Automata

**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**

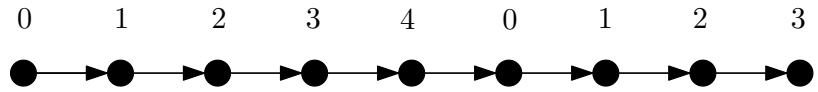
- Certify an orientation of the path.
- Check coherence locally.



# Paths & Automata

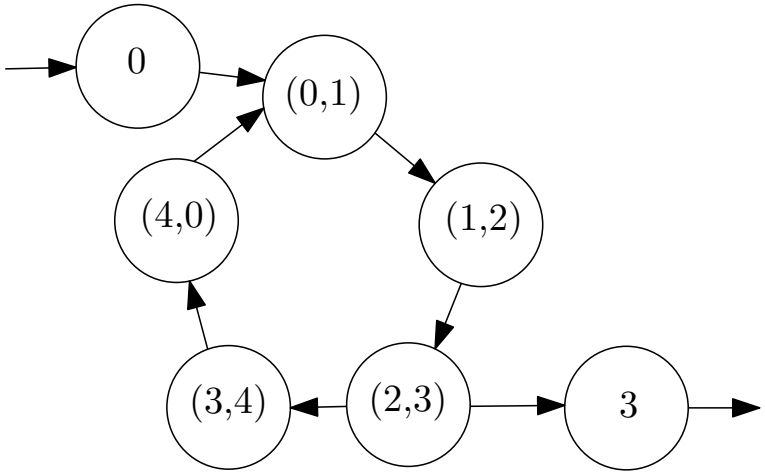
**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**




- Certify an orientation of the path.
- Check coherence locally.

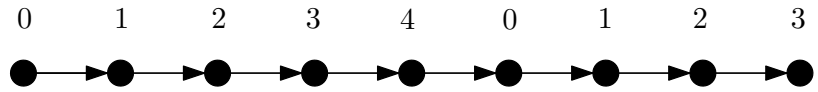
**Automata point of view :**



# Paths & Automata

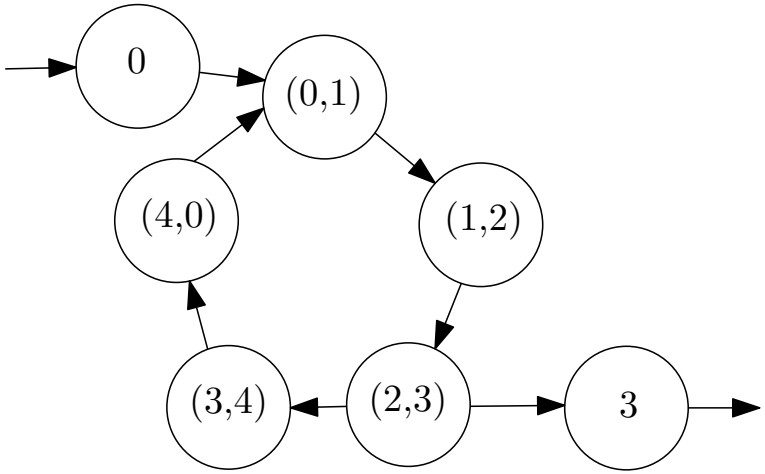
**Lemma** : Certify that a path has length $i$ mod $k$ can be done with $O(\log k)$ certificates.

**Proof**




- Certify an orientation of the path.
- Check coherence locally.

**Automata point of view :**




**Informal statement**

Certification with $r$ bits $\Leftrightarrow$ Automaton with $2^{2r}$ states.

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i)$

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i) \Leftrightarrow \mathcal{A}_k \cap (\overline{\cup_i^{k-1} \mathcal{A}_i})$

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i) \Leftrightarrow \mathcal{A}_k \cap (\overline{\cup_i^{k-1} \mathcal{A}_i})$

Automaton for $\mathcal{L}_k$ :

- (Union) $|\mathcal{A}_i| \leq 2^{2i} \Rightarrow |\cup_i^{k-1} \mathcal{A}_i| \leq 2^{2k}$.
- (Complementation) $|\overline{\cup_i^{k-1} \mathcal{A}_i}| \leq 2^{2^{2k}}$.
- (Intersection) $\mathcal{L}_k$ can be recognized with $2^{2^{2k}}$ states.

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i) \Leftrightarrow \mathcal{A}_k \cap \overline{(\cup_i^{k-1} \mathcal{A}_i)}$

Automaton for $\mathcal{L}_k$ :

- (Union) $|\mathcal{A}_i| \leq 2^{2i} \Rightarrow |\cup_i^{k-1} \mathcal{A}_i| \leq 2^{2k}$.
- (Complementation) $|\overline{\cup_i^{k-1} \mathcal{A}_i}| \leq 2^{2^{2k}}$.
- (Intersection) $\mathcal{L}_k$ can be recognized with $2^{2^{2k}}$ states.

$\mathcal{L}_k$ non empty $\Rightarrow$ $\mathcal{L}_k$ accepts a word of size $\leq 2^{2^{2k}}$

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i) \Leftrightarrow \mathcal{A}_k \cap \overline{(\cup_i^{k-1} \mathcal{A}_i)}$

Automaton for $\mathcal{L}_k$ :

- (Union) $|\mathcal{A}_i| \leq 2^{2i} \Rightarrow |\cup_i^{k-1} \mathcal{A}_i| \leq 2^{2k}$.
- (Complementation) $|\overline{\cup_i^{k-1} \mathcal{A}_i}| \leq 2^{2^{2k}}$.
- (Intersection) $\mathcal{L}_k$ can be recognized with $2^{2^{2k}}$ states.

$\mathcal{L}_k$ non empty $\Rightarrow \mathcal{L}_k$ accepts a word of size $\leq 2^{2^{2k}}$

$\Rightarrow \mathcal{L}_k$ empty if certification with less than $o(\log \log n)$ bits.

# Gap for paths

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log \log n)$ bits on anonymous paths $\Rightarrow$ Certification with $O(1)$ bits.

**Sketch of the proof :**

$\mathcal{A}_k$ : Automaton that recognizes all the paths using $\leq k$ bits.

$\mathcal{L}_k$ : Language recognized using $k$ bits but not $k-1$.

$\Leftrightarrow \mathcal{A}_k \setminus (\cup_i^{k-1} \mathcal{A}_i) \Leftrightarrow \mathcal{A}_k \cap (\overline{\cup_i^{k-1} \mathcal{A}_i})$

Automaton for $\mathcal{L}_k$ :

- (Union) $|\mathcal{A}_i| \leq 2^{2i} \Rightarrow |\cup_i^{k-1} \mathcal{A}_i| \leq 2^{2k}$.
- (Complementation) $|\overline{\cup_i^{k-1} \mathcal{A}_i}| \leq 2^{2^{2k}}$.
- (Intersection) $\mathcal{L}_k$ can be recognized with $2^{2^{2k}}$ states.

$\mathcal{L}_k$ non empty $\Rightarrow \mathcal{L}_k$ accepts a word of size $\leq 2^{2^{2k}}$

$\Rightarrow \mathcal{L}_k$ empty if certification with less than $o(\log \log n)$ bits.

**Remark :** Constant can be improved using Chrobak's theorem

# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

**Sketch of the proof :**

$p_i$ : $i$-th prime number

**Fact :**

$a_i := \prod_{j=1}^{i} p_i \approx 2^{i \log i}$ and $p_i \approx i \log i$.

# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

**Sketch of the proof :**

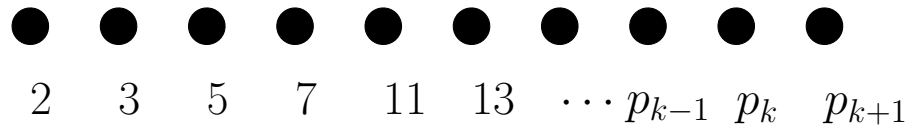
$p_i$ : $i$-th prime number

**Fact :**

$a_i := \prod_{j=1}^{i} p_i \approx 2^{i \log i}$ and $p_i \approx i \log i$.

**False good idea :**

1. $k$ : smallest integer such that $p_k | n$ and $p_{k-1} \nmid n$.
2. Certify $n$ mod $p_k$ and $p_{k-1}$.



# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

**Sketch of the proof :**

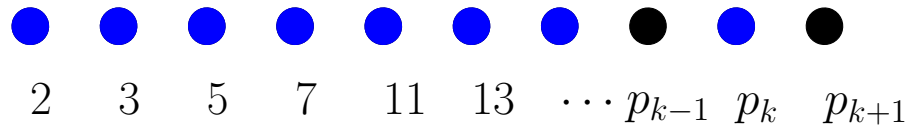
$p_i$ : $i$-th prime number

**Fact :**

$a_i := \prod_{j=1}^{i} p_i \approx 2^{i \log i}$ and $p_i \approx i \log i$.

**False good idea :**

1. $k$ : smallest integer such that $p_k | n$ and $p_{k-1} \nmid n$.
2. Certify $n$ mod $p_k$ and $p_{k-1}$.



# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

**Sketch of the proof :**

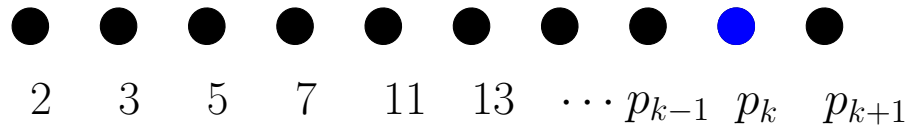
$p_i$ : $i$-th prime number

**Fact :**

$a_i := \prod_{j=1}^{i} p_i \approx 2^{i \log i}$ and $p_i \approx i \log i$.

**False good idea :**

1. $k$ : smallest integer such that $p_k | n$ and $p_{k-1} \nmid n$.
2. Certify $n$ mod $p_k$ and $p_{k-1}$.




Problem 1 : Might be too big

e.g. $p_k \approx \sqrt{n}$.

# Tightness

**Theorem**

$S = \{n \ / \ n$ is the product of the $i$ first prime (for some $i)\}$ can be certified with $O(\log \log n)$ bits.

**Sketch of the proof :**

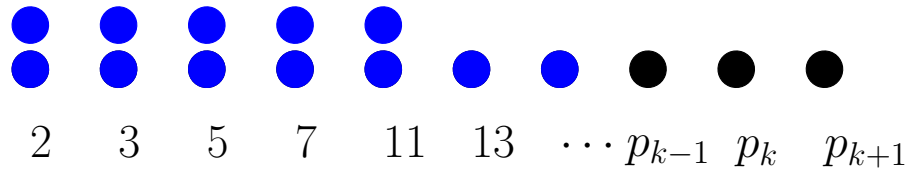
$p_i$ : $i$-th prime number

**Fact :**

$a_i := \prod_{j=1}^{i} p_i \approx 2^{i \log i}$ and $p_i \approx i \log i$.

**False good idea :**

1. $k$ : smallest integer such that $p_k | n$ and $p_{k-1} \nmid n$.
2. Certify $n$ mod $p_k$ and $p_{k-1}$.




Problem 1 : Might be too big

e.g. $p_k \approx \sqrt{n}$.

Problem 2 : Might not exist.

# Main lemma

**Lemma :**

For $n \geq 3$, $n \notin S$ iff :

1. $n$ is odd or,
2. $\exists p_k \leq p_{k+1} \leq c \log n$ such that $p_{k+1} | n$ and $p_k \nmid n$ or,

# Main lemma

**Lemma :**

For $n \geq 3$, $n \notin S$ iff :

1. $n$ is odd or,

2. $\exists p_k \leq p_{k+1} \leq c \log n$ such that $p_{k+1} | n$ and $p_k \nmid n$ or,

3. $\exists p_k \leq c \log n$ and $m \leq \log n$ such that $p_k | n$, $p_{k+1} \nmid n$ and $n \neq a_k := \Pi_{i=1}^{k} p_i \mod m$

(Chinese remainder theorem)

# Gap for cycles

**Theorem** (B., Feuilloley, Zeitoun '25+)

Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

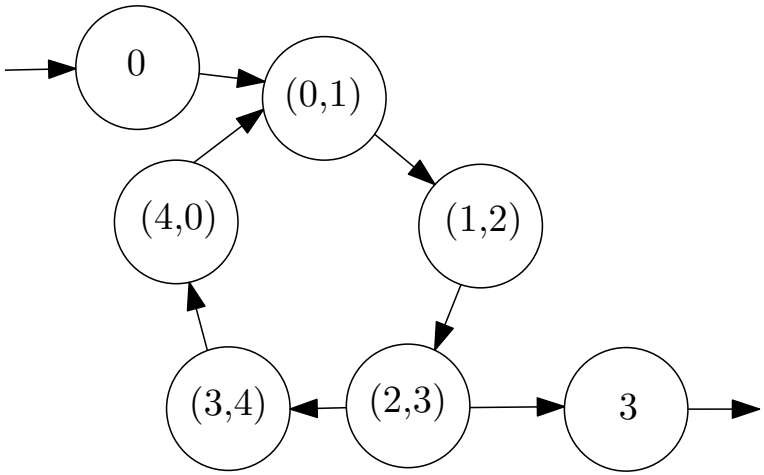
# Gap for cycles

**Theorem** (B., Feuilloley, Zeitoun '25+)

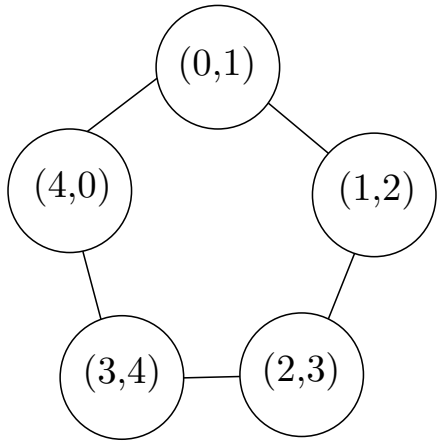
Certification with $o(\log n)$ bits on anonymous cycles $\Rightarrow$ Certification with $O(1)$ bits.

## What is different ?

Paths




Cycles



# Graph of certification

$S$ : a property $\Leftrightarrow$ Subset of $\mathbb{N}$

$S_k$ : Subset of $S$ recognized with $\leq k$ bits.

# Graph of certification

$S$ : a property $\Leftrightarrow$ Subset of $\mathbb{N}$
$S_k$ : Subset of $S$ recognized with $\leq k$ bits.

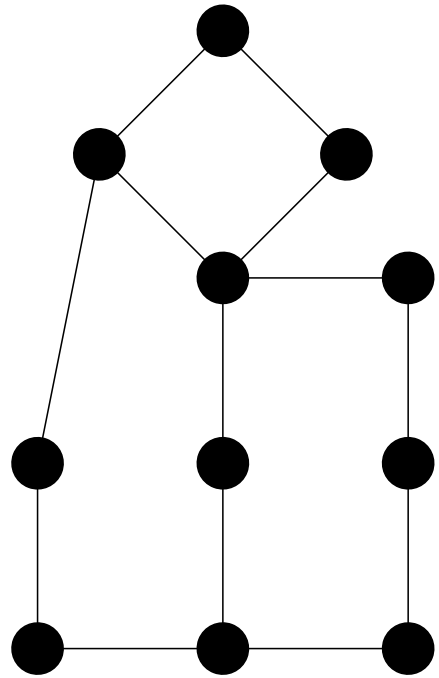
$\mathcal{G}_k$ defined by

- Vertices : pairs of certificates of $\leq k$ bits $\Leftrightarrow \leq 2^{2k}$ vertices
- Edges : $(a, b) - (b, c)$ if there exists an accepting run where two adjacent vertices receive these certificates.

**Remark :**
$n$ accepted with $\leq k$ bits $(n \in S_k) \Leftrightarrow$ Cycle length $n$ in $\mathcal{G}_k$.

# Sketch of the proof

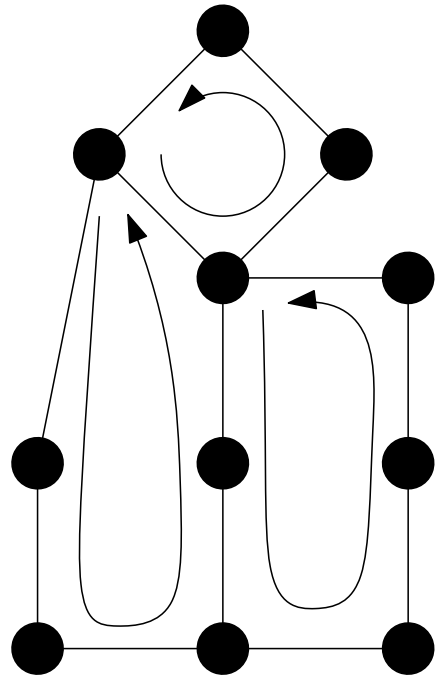
- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.



# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

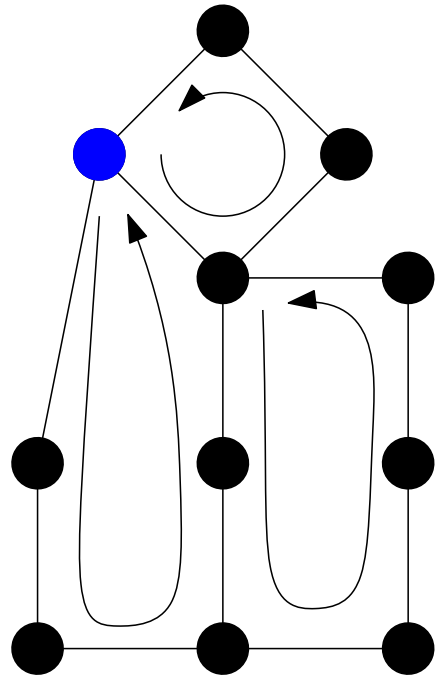
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } {}_{\text{(a conn. comp. of)}}\ \mathcal{G}_k)$.

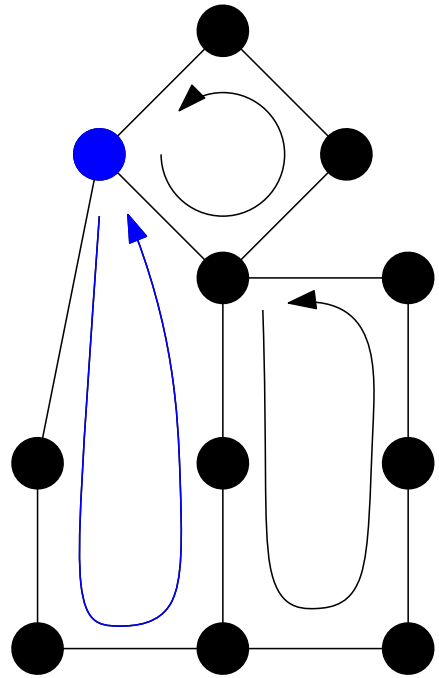
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } {}_{\text{(a conn. comp. of)}}\ \mathcal{G}_k)$.

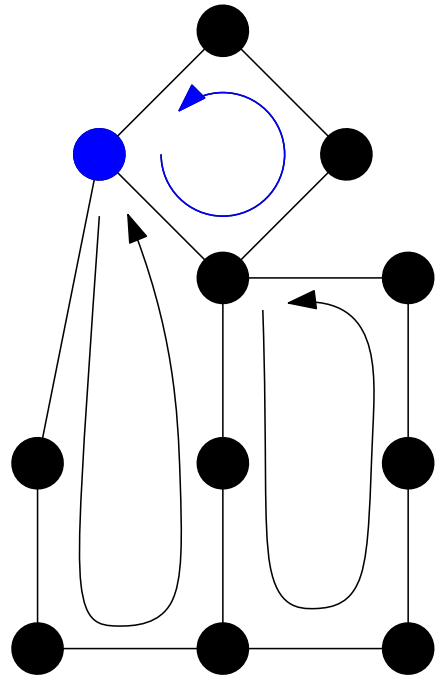
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

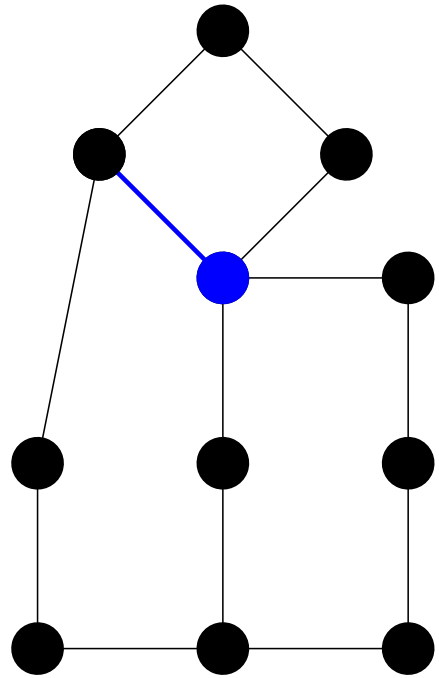
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } {}_{\text{(a conn. comp. of)}}\ \mathcal{G}_k)$.

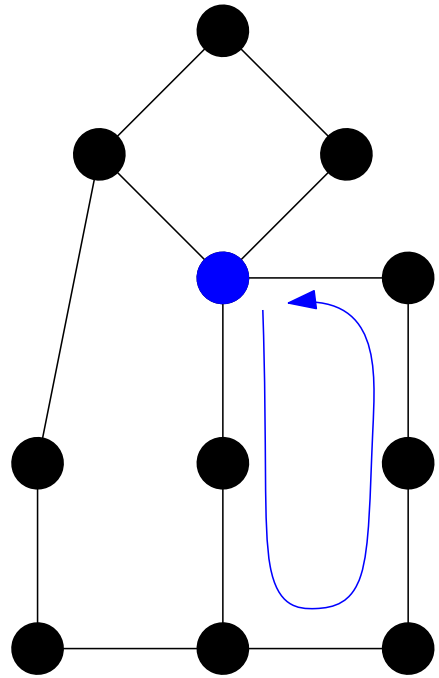
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

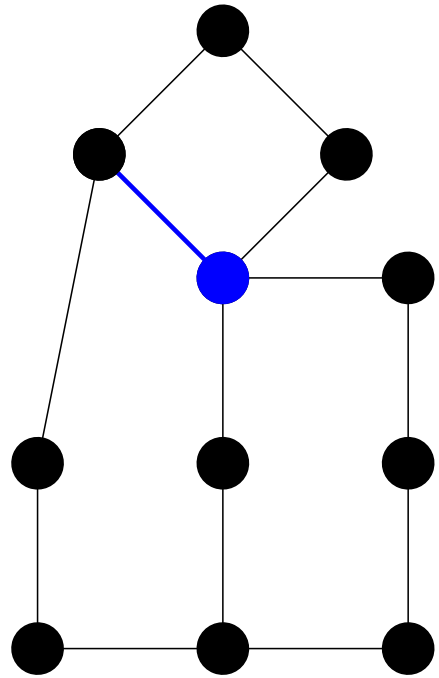
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } {}_{\text{(a conn. comp. of)}}\ \mathcal{G}_k)$.

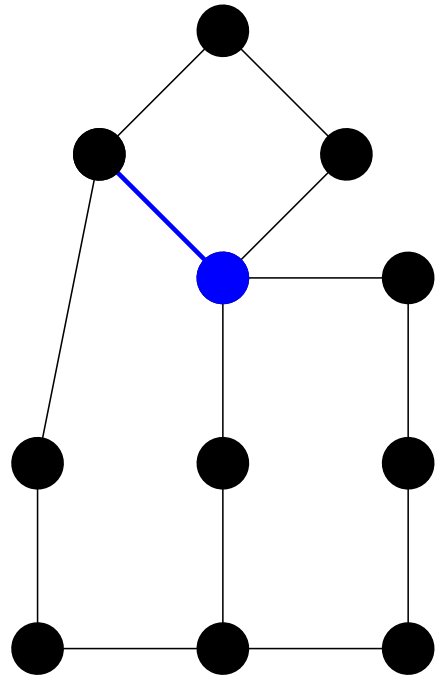
$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles



**Tool 1.** (Generalized Bezout theorem) $d = gcd(n_1, \ldots, n_k)$. If $n \geq \max(n_i)^2$ and $d|n$ then, $n = \sum a_i n_i$ where $a_i \geq 0$

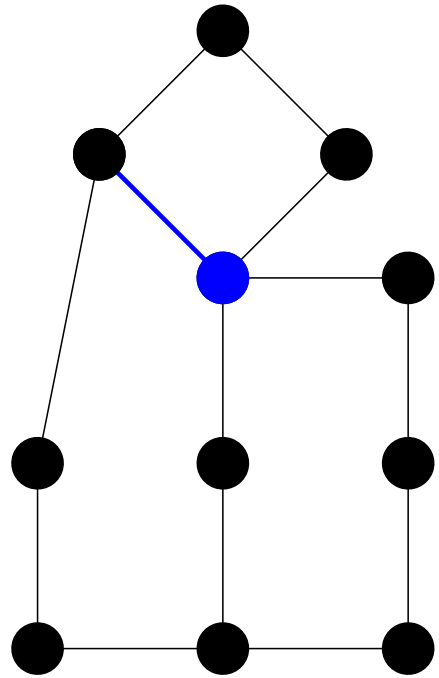
Classic version : $\forall n, m \in \mathbb{N}$, $\exists a, b \in \mathbb{Z}$ such that $na + mb = gcd(n, m)$.

# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles

- $m \geq 2^{\Omega(k)}$ and $d|m \Rightarrow m \in S_k$



**Tool 1.** (Generalized Bezout theorem) $d = gcd(n_1, \ldots, n_k)$. If $n \geq \max(n_i)^2$ and $d|n$ then, $n = \sum a_i n_i$ where $a_i \geq 0$

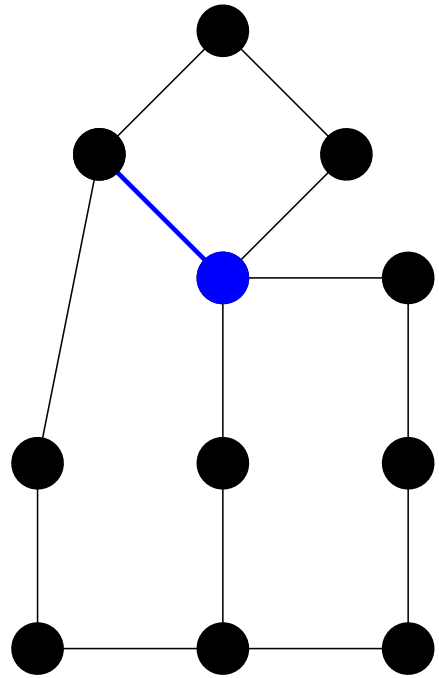
Classic version : $\forall n, m \in \mathbb{N}$, $\exists a, b \in \mathbb{Z}$ such that $na + mb = gcd(n, m)$.

# Sketch of the proof

- $d = gcd(\text{cycles in } \text{(a conn. comp. of) } \mathcal{G}_k)$.

$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles

- $m \geq 2^{\Omega(k)}$ and $d|m \Rightarrow m \in S_k$
- If $n >> 2^{\Omega(k)} \Rightarrow \exists$ many primes between $2^{\Omega(k)}$ and $n$.



**Tool 1.** (Generalized Bezout theorem) $d = gcd(n_1, \ldots, n_k)$. If $n \geq \max(n_i)^2$ and $d|n$ then, $n = \sum a_i n_i$ where $a_i \geq 0$

Classic version : $\forall n, m \in \mathbb{N}$, $\exists a, b \in \mathbb{Z}$ such that $na + mb = gcd(n, m)$.

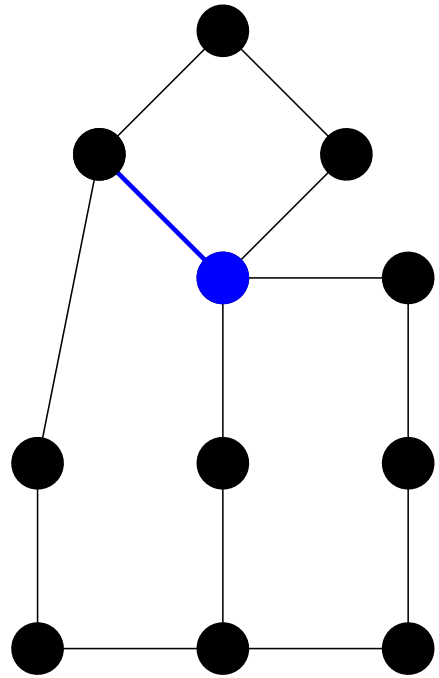
**Tool 2.** Prime numbers are "dense"

# Sketch of the proof

- $d = gcd$(cycles in (a conn. comp. of) $\mathcal{G}_k$).

$\Rightarrow \exists$ closed walk of length $\leq O(2^{O(k)})$ passing through all the cycles

- $m \geq 2^{\Omega(k)}$ and $d|m \Rightarrow m \in S_k$
- If $n >> 2^{\Omega(k)} \Rightarrow \exists$ many primes between $2^{\Omega(k)}$ and $n$.
- If min. $n \in S_k \setminus (\cup_{i \leq k} S_i)$ is too large w.r.t $2^{\Omega(k)} \Rightarrow$ Contradiction



**Tool 1.** (Generalized Bezout theorem) $d = gcd(n_1, \ldots, n_k)$. If $n \geq \max(n_i)^2$ and $d|n$ then, $n = \sum a_i n_i$ where $a_i \geq 0$

Classic version : $\forall n, m \in \mathbb{N}$, $\exists a, b \in \mathbb{Z}$ such that $na + mb = gcd(n, m)$.

**Tool 2.** Prime numbers are "dense"

# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

# No gap for caterpillar

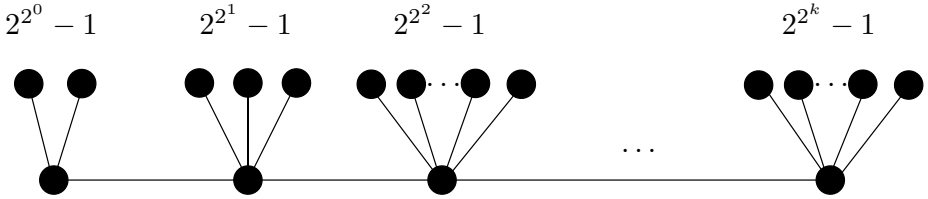
**Theorem** (B., Feuilloley, Zeitoun 25+)

No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

## Proof on an example :

- $n = \sum_{i=0}^{k} 2^{2^i}$



# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

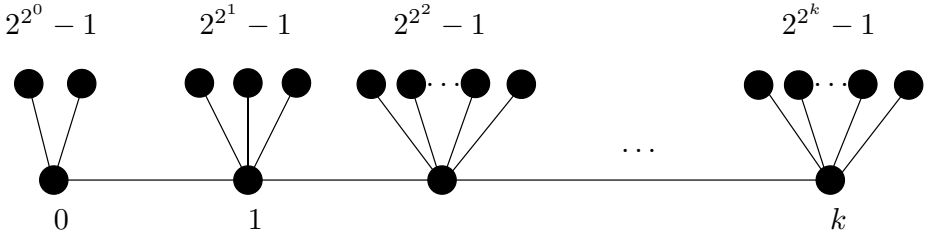
No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

## Proof on an example :

- $n = \sum_{i=0}^{k} 2^{2^i}$
- Certified with $\log k$ bits

$\rightarrow O(\log \log \log n)$



# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

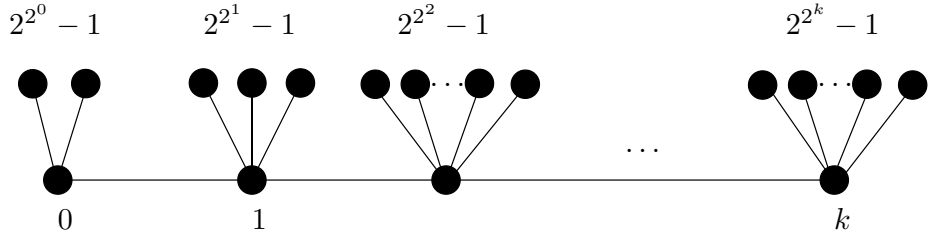
No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

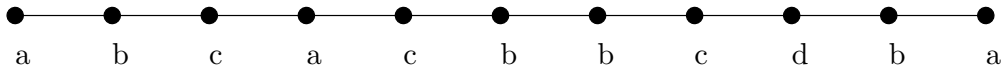
## Proof on an example :

- $n = \sum_{i=0}^{k} 2^{2^i}$
- Certified with $\log k$ bits

$\rightarrow O(\log \log \log n)$




No certification with a constant number of bits :



# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

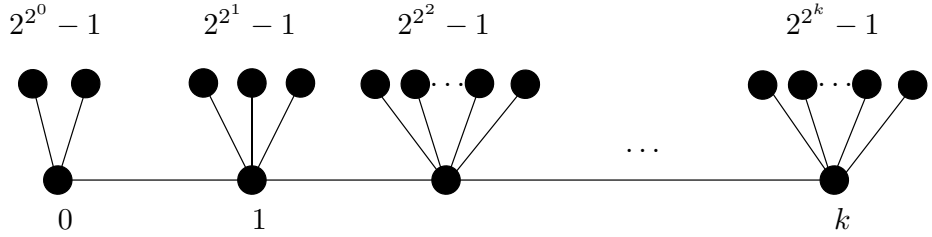
No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

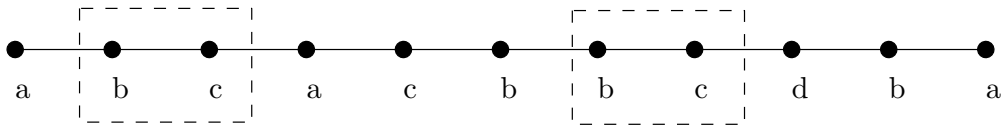
## Proof on an example :

- $n = \sum_{i=0}^{k} 2^{2^i}$
- Certified with $\log k$ bits

$\rightarrow O(\log \log \log n)$




No certification with a constant number of bits :



# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

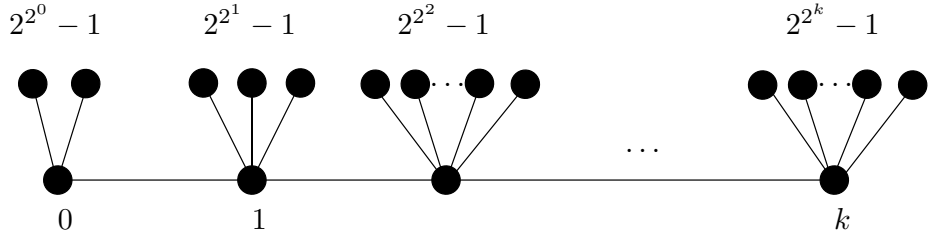
No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

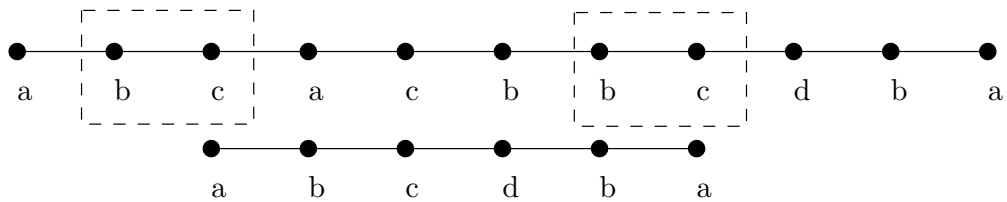
## Proof on an example :

- $n = \sum_{i=0}^{k} 2^{2^i}$
- Certified with $\log k$ bits

$\rightarrow O(\log \log \log n)$




No certification with a constant number of bits :



# No gap for caterpillar

**Theorem** (B., Feuilloley, Zeitoun 25+)

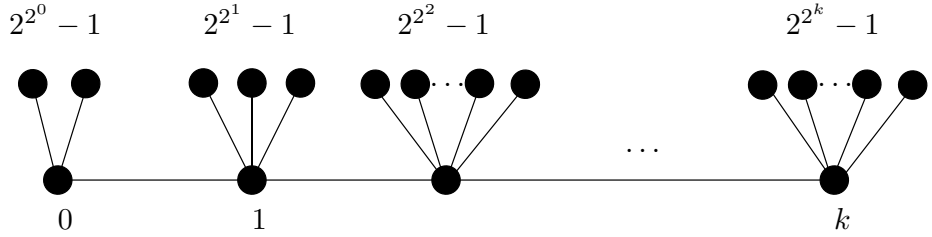
No gap for caterpillars (in terms of $n$).

(For every $f$, there exists a property that can be certified with $f(n)$ bits but not with $o(f(n))$)

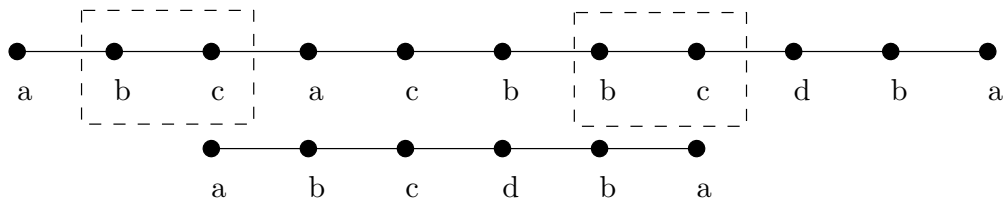
**Proof on an example :**

- $n = \sum_{i=0}^{k} 2^{2^i}$
- Certified with $\log k$ bits

$\rightarrow O(\log \log \log n)$




No certification with a constant number of bits :




**Remark :** We can close the gap between lower and upper bound.

# Open problems

**Question.**

What about between $\log \log n$ and $\log n$ ?

We can find properties we can certify with functions "in the middle" but we cannot prove tightness of the LB.

# Open problems

**Question.**

What about between $\log \log n$ and $\log n$ ?

We can find properties we can certify with functions "in the middle" but we cannot prove tightness of the LB.

**Question.**

Non-anonymous model ? Model where only $v$ can see its ID ?

# Open problems

**Question.**

What about between $\log \log n$ and $\log n$ ?

We can find properties we can certify with functions "in the middle" but we cannot prove tightness of the LB.

**Question.**

Non-anonymous model ? Model where only $v$ can see its ID ?

**Question 2.**

General graphs of bounded degree ?

# Open problems

**Question.**
What about between $\log \log n$ and $\log n$ ?

We can find properties we can certify with functions "in the middle" but we cannot prove tightness of the LB.

**Question.**
Non-anonymous model ? Model where only $v$ can see its ID ?

**Question 2.**
General graphs of bounded degree ?

**Question 3.**
What if nodes have access to an approximation of $n$ ?

Very partial results

# Open problems

**Question.**
What about between $\log \log n$ and $\log n$ ?

We can find properties we can certify with functions "in the middle" but we cannot prove tightness of the LB.

**Question.**
Non-anonymous model ? Model where only $v$ can see its ID ?

**Question 2.**
General graphs of bounded degree ?

**Question 3.**
What if nodes have access to an approximation of $n$ ?

Very partial results

**Thanks for your attention !**