# Monitoring edge-geodetic sets of chordal graphs

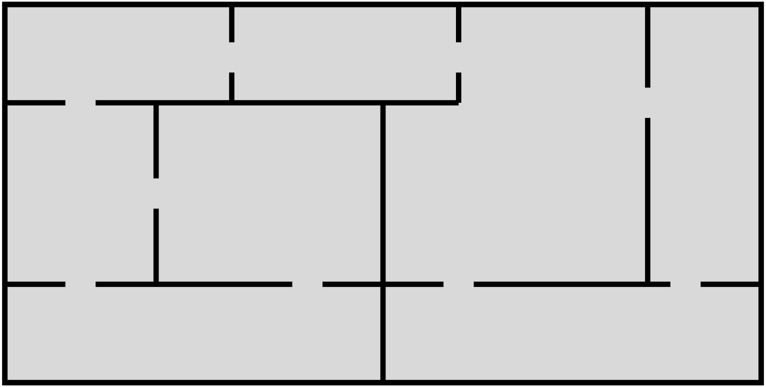
Clara Marcille[1] Nacim Oijid[2]

[1]LIP, ENS de Lyon

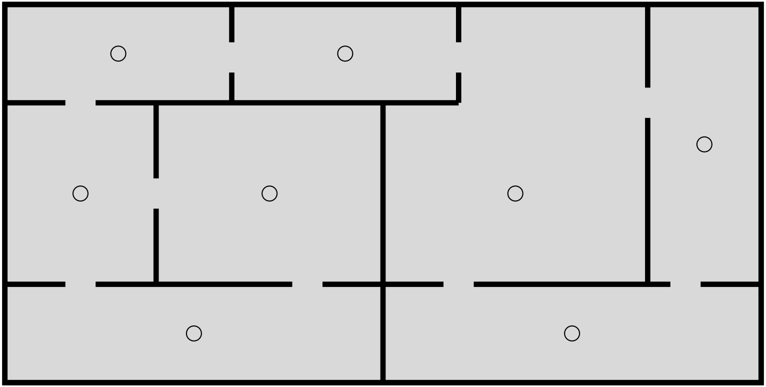
[2]Umeå University

IRIF Graph Seminar, 27 January 2026

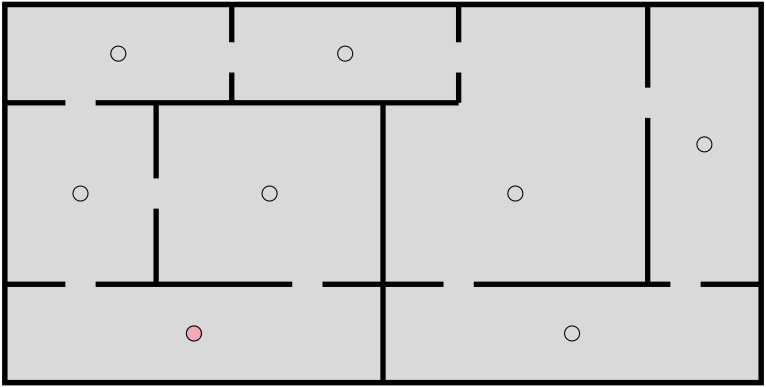
# Detecting Events in Graphs



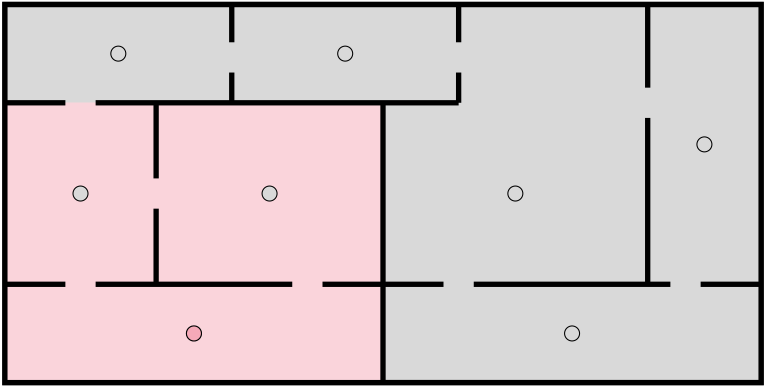
# Detecting Events in Graphs



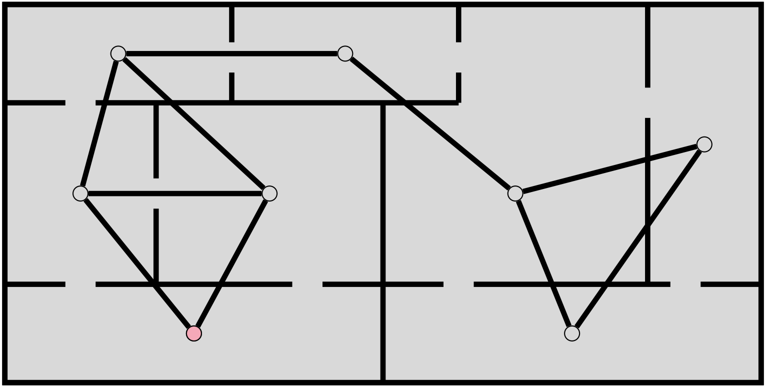
# Detecting Events in Graphs



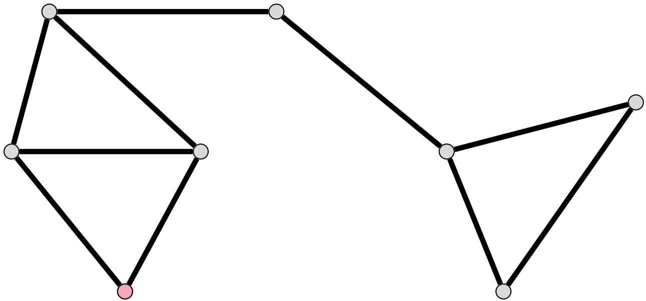
# Detecting Events in Graphs



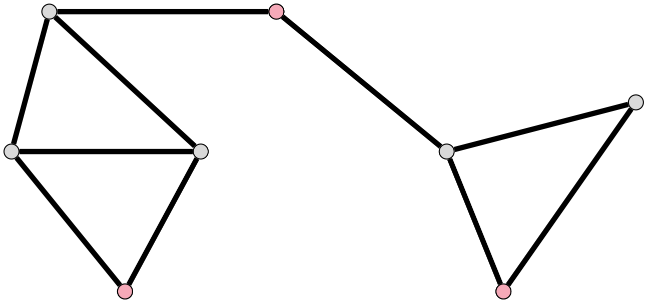
# Detecting Events in Graphs



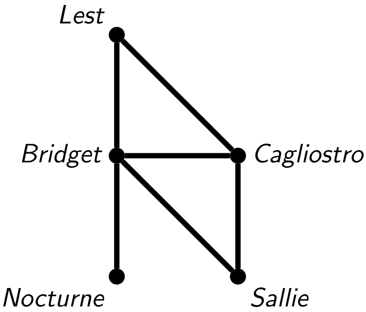
# Detecting Events in Graphs



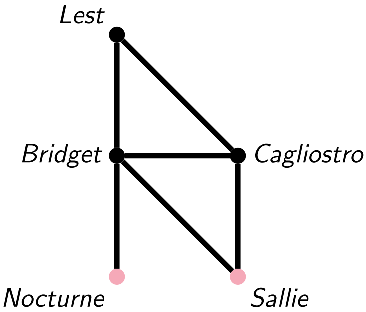
# Detecting Events in Graphs



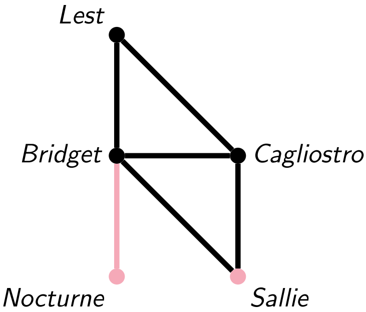
# Monitoring Networks through Distances



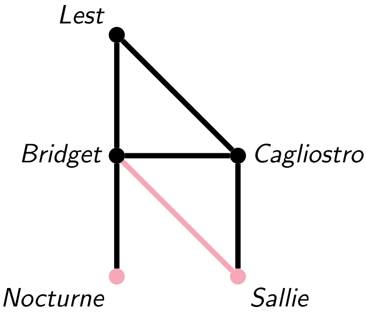
# Monitoring Networks through Distances



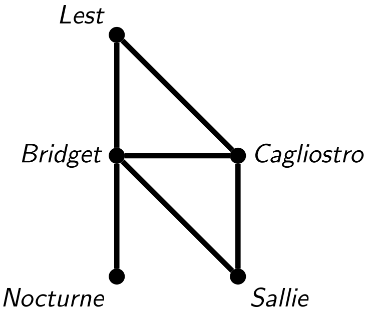
# Monitoring Networks through Distances



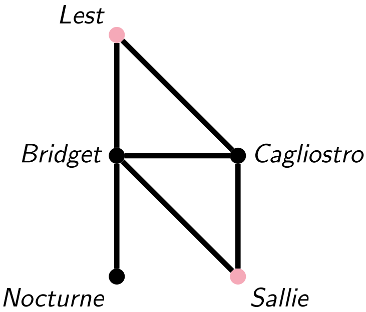
# Monitoring Networks through Distances



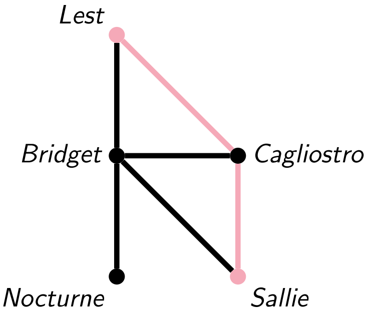
# Monitoring Networks through Distances



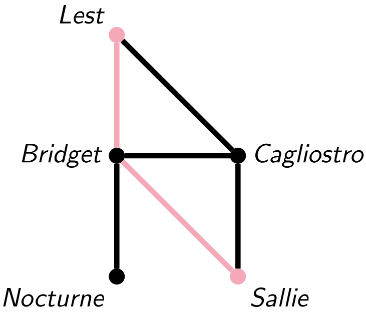
# Monitoring Networks through Distances



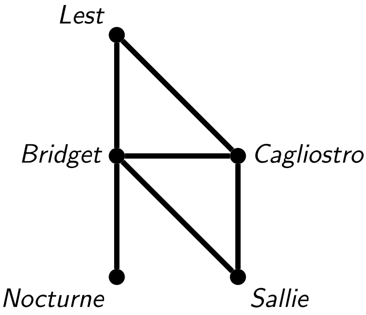
# Monitoring Networks through Distances



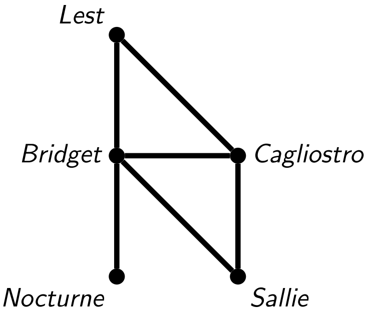
# Monitoring Networks through Distances



# Monitoring Networks through Distances



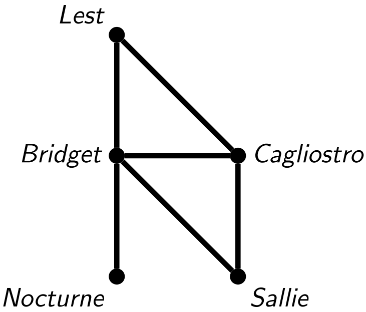
# Monitoring Networks through Distances




We make the following hypotheses:

- messages have a timestamp;
- every step takes the same time;
- messages take a shortest path;
- everyone knows their distance to others.

# Monitoring Networks through Distances



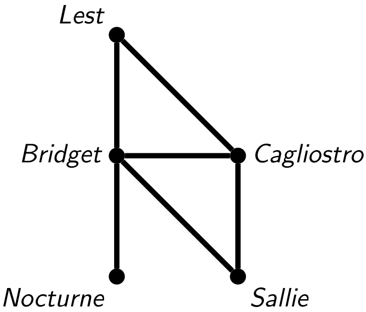

We make the following hypotheses:

- messages have a timestamp;
- every step takes the same time;
- messages take a shortest path;
- everyone knows their distance to others.

**Link with networks**

This object simulates probes monitoring a network: if the value of the ping between two probes increases, then one can know a failure happened.

# Monitoring Networks through Distances




We make the following hypotheses:

- messages have a timestamp;
- every step takes the same time;
- messages take a shortest path;
- everyone knows their distance to others.

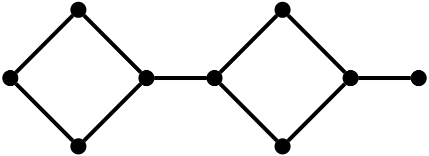
## Link with networks

This object simulates probes monitoring a network: if the value of the ping between two probes increases, then one can know a failure happened.

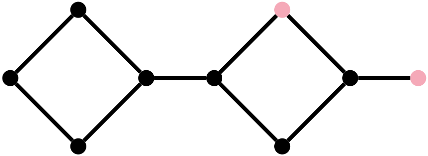
## Goal

We want to minimise the number of probes.

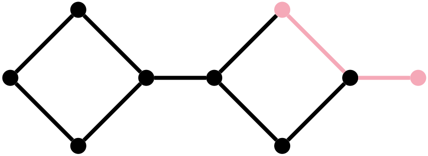
# Monitoring Edge Geodetics



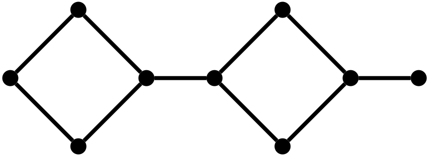
# Monitoring Edge Geodetics



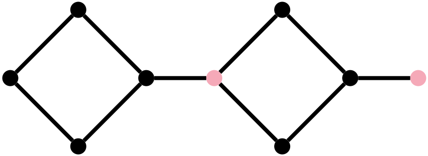
# Monitoring Edge Geodetics



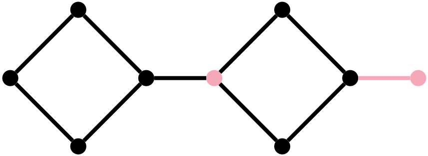
# Monitoring Edge Geodetics



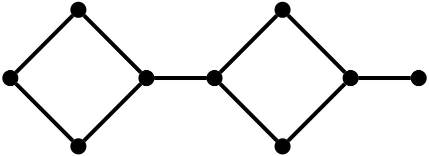
# Monitoring Edge Geodetics



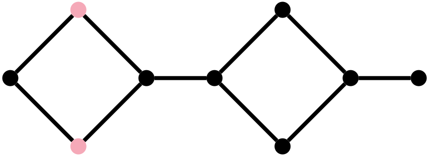
# Monitoring Edge Geodetics



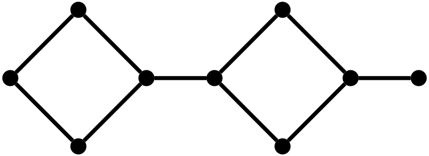
# Monitoring Edge Geodetics



# Monitoring Edge Geodetics



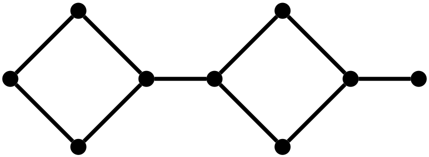
# Monitoring Edge Geodetics



**Definition (MEG-set) [FNRS23]**

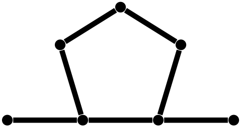
A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.

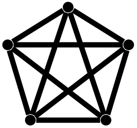
# Monitoring Edge Geodetics



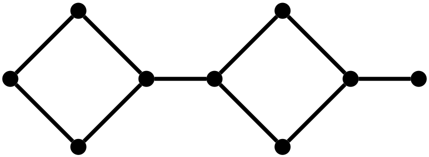
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



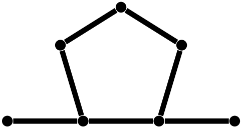


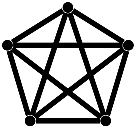
# Monitoring Edge Geodetics



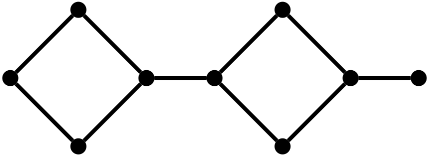
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



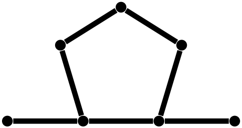


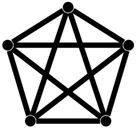
# Monitoring Edge Geodetics



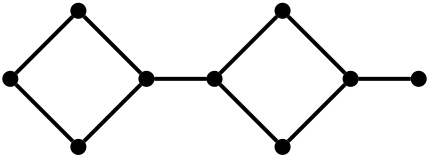
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



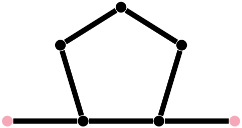


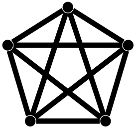
# Monitoring Edge Geodetics



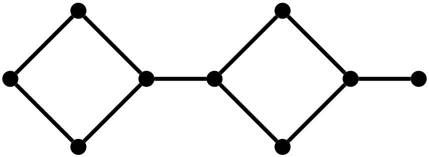
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



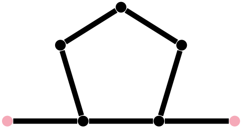


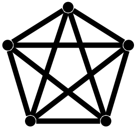
# Monitoring Edge Geodetics



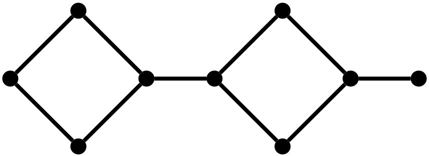
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



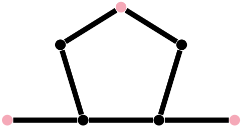


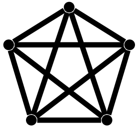
# Monitoring Edge Geodetics



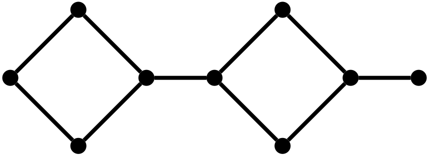
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



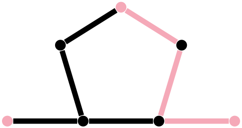


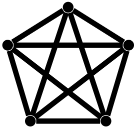
# Monitoring Edge Geodetics



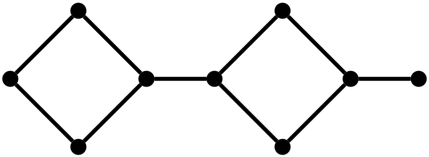
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



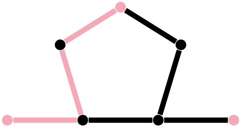


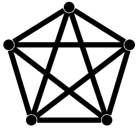
# Monitoring Edge Geodetics



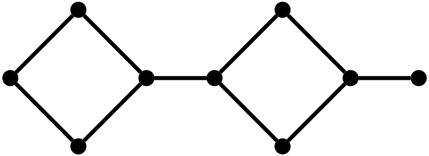
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



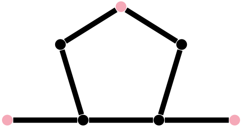


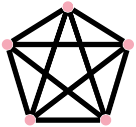
# Monitoring Edge Geodetics



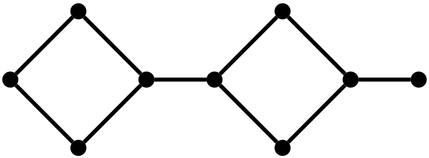
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



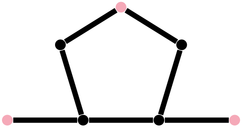


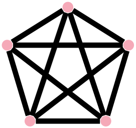
# Monitoring Edge Geodetics



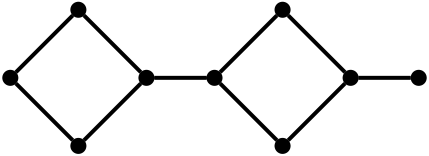
**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.



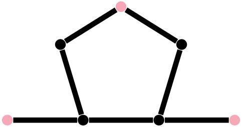


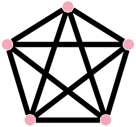
# Monitoring Edge Geodetics



**Definition (MEG-set) [FNRS23]**

A set $M$ of vertices *monitors* an edge $e$ if $e$ lies on all shortest paths between two vertices of $M$.





**Theorem [Haslegrave, 2023]**

Deciding for a graph $G$ and a natural number $k$ whether $meg(G) \leq k$ is NP-complete.

# Complexity Aspects of MEG-sets

**Theorem [Haslegrave, 2023]**

Deciding for a graph $G$ and a natural number $k$ whether $meg(G) \leq k$ is NP-complete.

# Complexity Aspects of MEG-sets

**Theorem [Haslegrave, 2023]**

Deciding for a graph $G$ and a natural number $k$ whether $meg(G) \leq k$ is NP-complete.

Previous results ([FMSST25]):

- MEG-SET is XP by solution size.
- MEG-SET is FPT by clique-width plus diameter.
- MEG-SET is FPT by tree-width on chordal graphs.

## Parameterised Complexity

# Complexity Aspects of MEG-sets

**Theorem [Haslegrave, 2023]**

Deciding for a graph $G$ and a natural number $k$ whether $meg(G) \leq k$ is NP-complete.

Previous results ([FMSST25]):

- MEG-SET is XP by solution size.
- MEG-SET is FPT by clique-width plus diameter.
- MEG-SET is FPT by tree-width on chordal graphs.

Parameterised Complexity

For a graph $G$, there is a polynomial-time algorithm to:

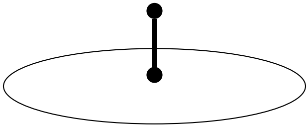
- find a MEG-set of $G$ of size $meg(G) \cdot \sqrt{n \ln m}$, but
- not of size $c \cdot meg(G)$ for any $c > 1$.

Approximation Algorithms

# Mandatory Vertices

**Definition**

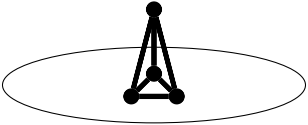
A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.



# Mandatory Vertices

**Definition**

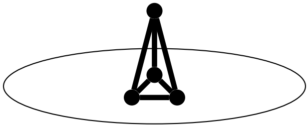
A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

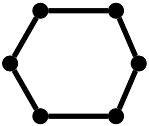


# Mandatory Vertices

**Definition**

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

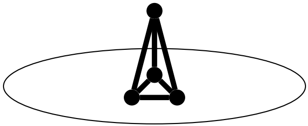


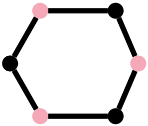


# Mandatory Vertices

## Definition

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

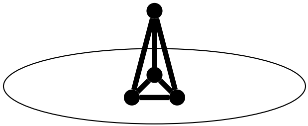


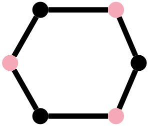


# Mandatory Vertices

**Definition**

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

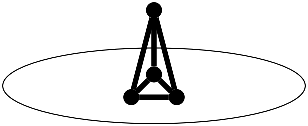


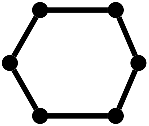


# Mandatory Vertices

**Definition**

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.





# Mandatory Vertices

**Definition**

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

**Theorem [FMMSST24]**

We can find the set of mandatory vertices of a graph in polynomial time.

**Definition**

A graph is *MEG-minimal* if the set of mandatory vertices is a MEG-set.

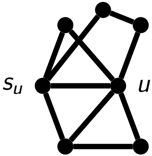
# Mandatory Vertices

**Definition**

A vertex of $G$ is *mandatory* if it belongs to all MEG-sets of $G$.

**Theorem [FMMSST24]**

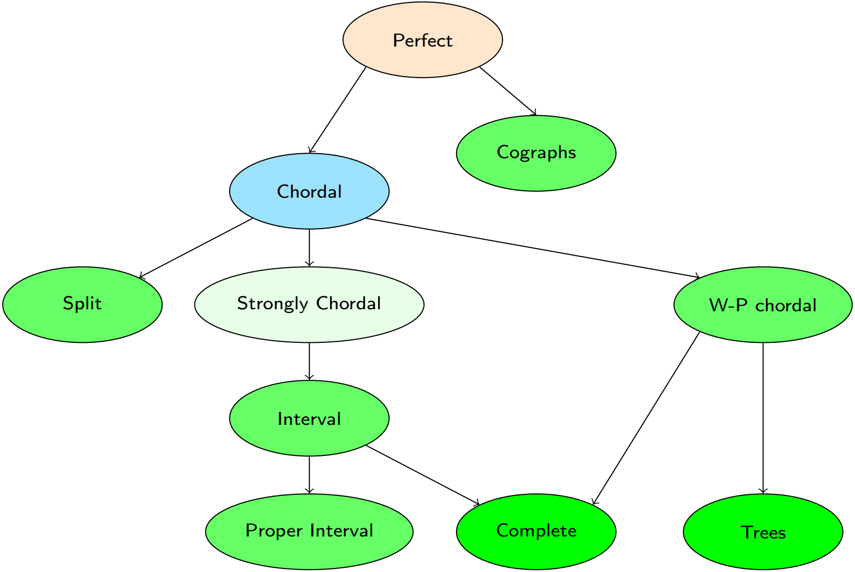
We can find the set of mandatory vertices of a graph in polynomial time.




**Definition**

A graph is *MEG-minimal* if the set of mandatory vertices is a MEG-set.

# MEG-minimal Graph Classes



# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

**Main Theorem [FMMSST24]**

The set of mandatory vertices of an interval graph is a MEG-set.

# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

**Main Theorem [FMMSST24]**

The set of mandatory vertices of an interval graph is a MEG-set.

# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

**Main Theorem [FMMSST24]**

The set of mandatory vertices of an interval graph is a MEG-set.

# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

**Main Theorem [FMMSST24]**

The set of mandatory vertices of an interval graph is a MEG-set.

# Idea of the Interval Graphs Result

**Lemma [FMMSST24]**

In an interval graph, a vertex is mandatory if and only if its neighbourhood has diameter at most 4.

**Lemma [FMMSST24]**

All interval graphs have a representation where the leftmost and rightmost intervals are mandatory vertices.

**Main Theorem [FMMSST24]**

The set of mandatory vertices of an interval graph is a MEG-set.

# Chordal Graphs

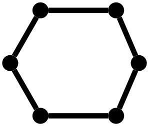
**Definition**

A graph is *chordal* if every cycle of size at least 4 has a chord.

# Chordal Graphs

**Definition**

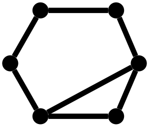
A graph is *chordal* if every cycle of size at least 4 has a chord.



# Chordal Graphs

### Definition

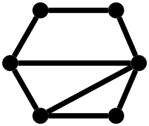
A graph is *chordal* if every cycle of size at least 4 has a chord.



# Chordal Graphs

**Definition**

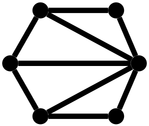
A graph is *chordal* if every cycle of size at least 4 has a chord.



# Chordal Graphs

## Definition

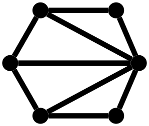
A graph is *chordal* if every cycle of size at least 4 has a chord.



# Chordal Graphs

**Definition**

A graph is *chordal* if every cycle of size at least 4 has a chord.



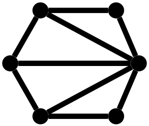
This is equivalent to:

- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.

# Chordal Graphs

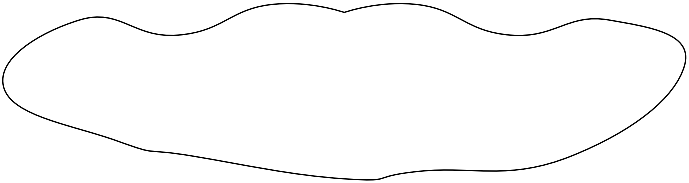
**Definition**

A graph is *chordal* if every cycle of size at least 4 has a chord.



This is equivalent to:

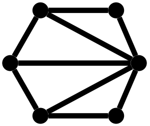
- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.



# Chordal Graphs

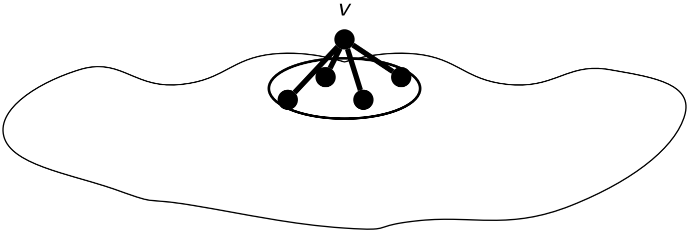
**Definition**

A graph is *chordal* if every cycle of size at least 4 has a chord.



This is equivalent to:

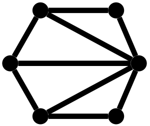
- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.



# Chordal Graphs

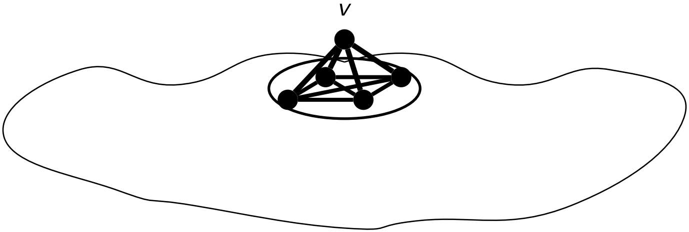
## Definition

A graph is *chordal* if every cycle of size at least 4 has a chord.



This is equivalent to:

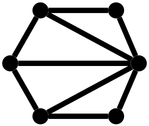
- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.



# Chordal Graphs

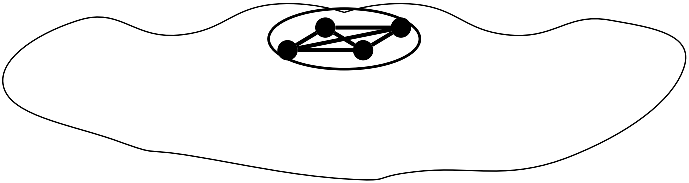
## Definition

A graph is *chordal* if every cycle of size at least 4 has a chord.



This is equivalent to:

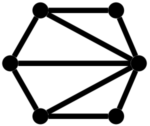
- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.



# Chordal Graphs

**Definition**

A graph is *chordal* if every cycle of size at least 4 has a chord.



This is equivalent to:

- having no induced $C_k$ for all $k \geq 4$;
- admitting an elimination scheme of simplicial vertices.

**Theorem [FNRS23]**

Simplicial vertices are mandatory.

# Overview of the Proof

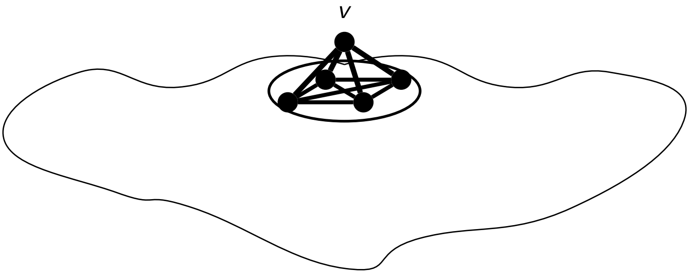
## Idea of the proof

- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.

# Overview of the Proof

## Idea of the proof

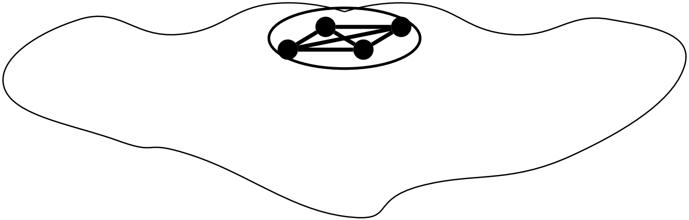
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

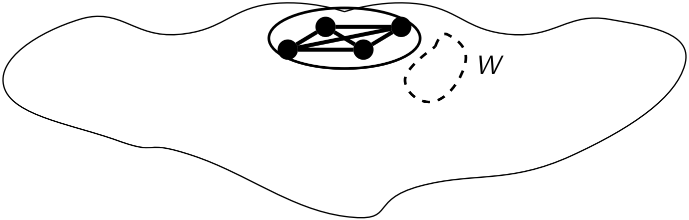
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

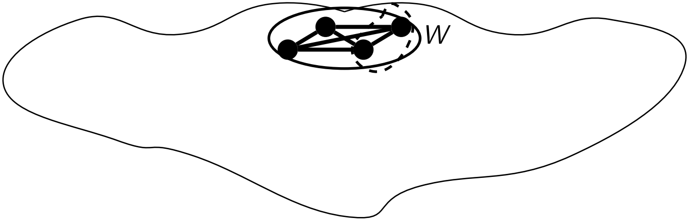
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

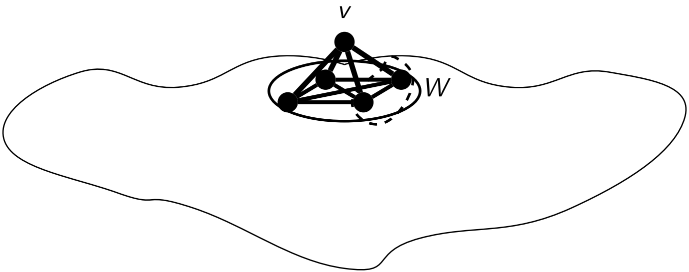
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

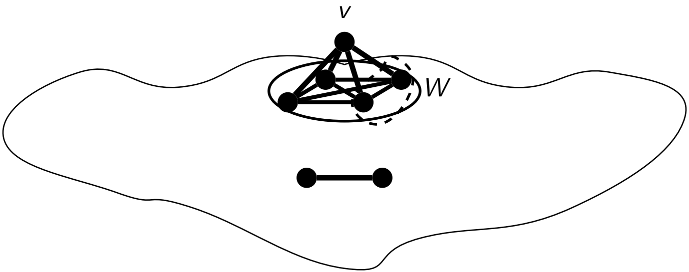
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

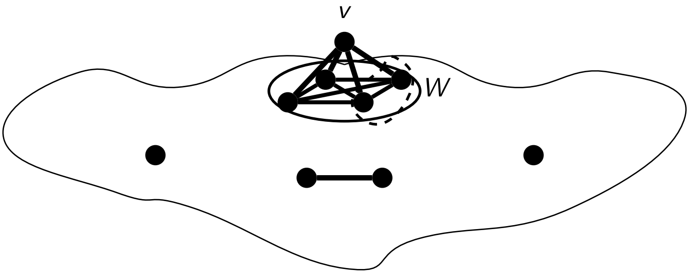
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

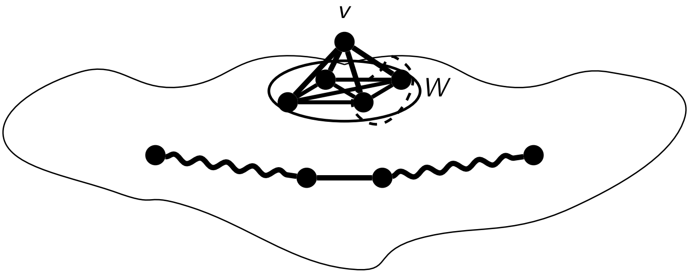
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

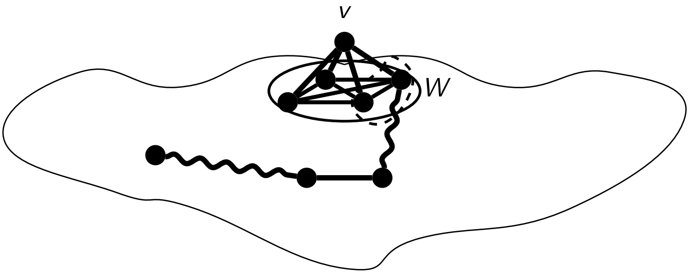
- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# Overview of the Proof

## Idea of the proof

- Consider a minimal counterexample $G$ and remove $v$ the first vertex of an elimination scheme of $G$.
- Look at $W$ the vertices mandatory in $G - v$ but not in $G$, and prove $W \subset N(v)$.
- Show that all the edges of $G - v$ monitored (in $G - v$) are still monitored in $G$.
- Show that all the edges incident to $v$ are monitored.



# $W \subset N(v)$

Theorem [FMMSST24]

A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.

# $W \subset N(v)$

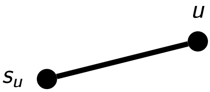
**Theorem [FMMSST24]**

A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.

# $W \subset N(v)$

**Theorem [FMMSST24]**

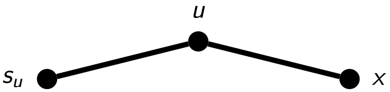
A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.



# $W \subset N(v)$

**Theorem [FMMSST24]**

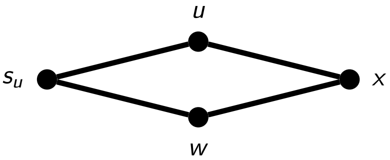
A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.



# $W \subset N(v)$

**Theorem [FMMSST24]**

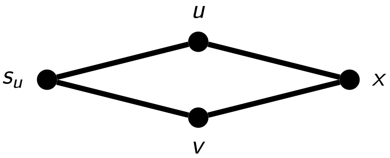
A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.



# $W \subset N(v)$

**Theorem [FMMSST24]**

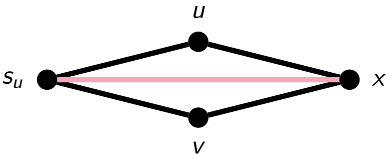
A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.



# $W \subset N(v)$

**Theorem [FMMSST24]**

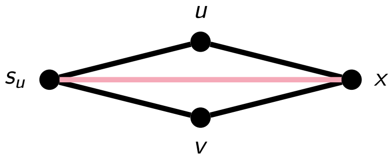
A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.



# $W \subset N(v)$

**Theorem [FMMSST24]**

A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u u x$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u u x w$ is a cycle.




**Corollary**

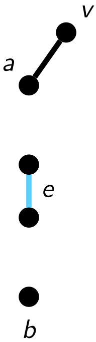
We look at the edges monitored by a vertex of $W$.

# Monitoring in the General Case

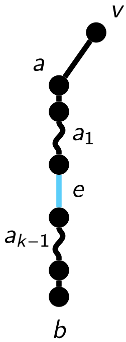
# Monitoring in the General Case

# Monitoring in the General Case

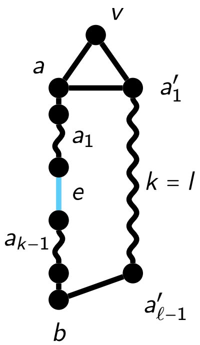
# Monitoring in the General Case



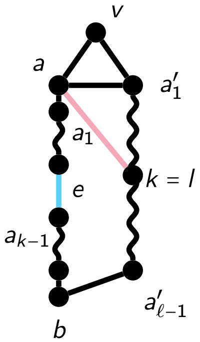
# Monitoring in the General Case



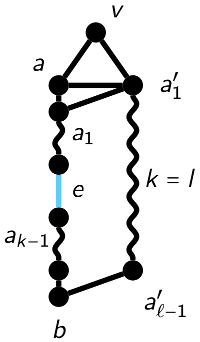
# Monitoring in the General Case



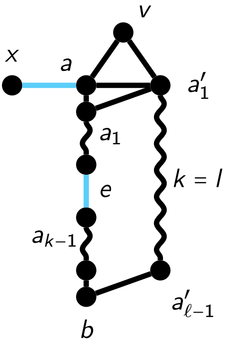
# Monitoring in the General Case



# Monitoring in the General Case



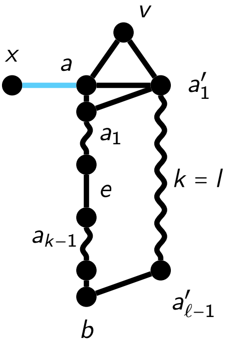
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $u$ is *mandatory* if there exists $s_u$ such that for any $x \in N(u)$ such that $s_u ux$ is an induced 2-path, there exists $w \in V(G)$ such that $s_u uxw$ is a cycle.

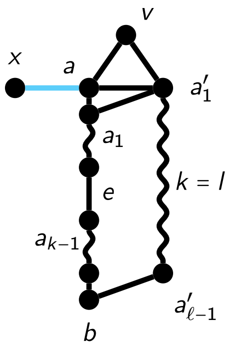
# Monitoring in the General Case




Theorem [FMMSST24]

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

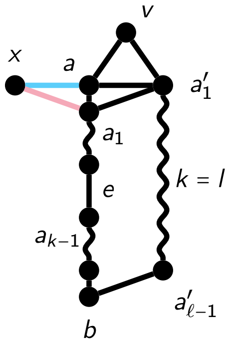
# Monitoring in the General Case




Theorem [FMMSST24]

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

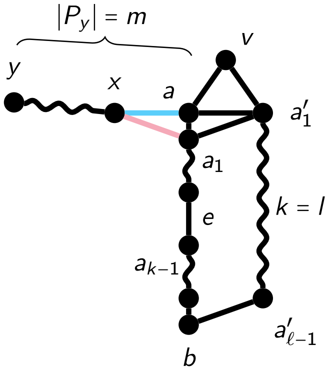
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1 a x$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1 a x z$ is a cycle.

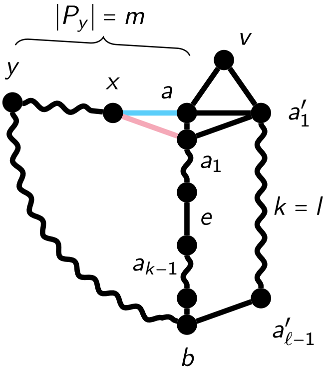
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

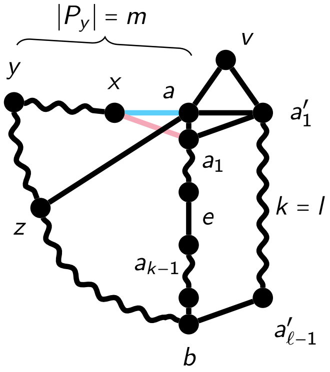
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

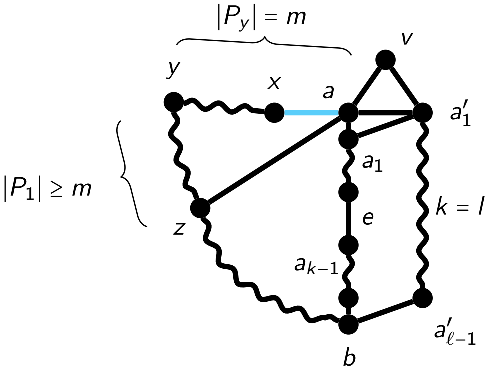
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

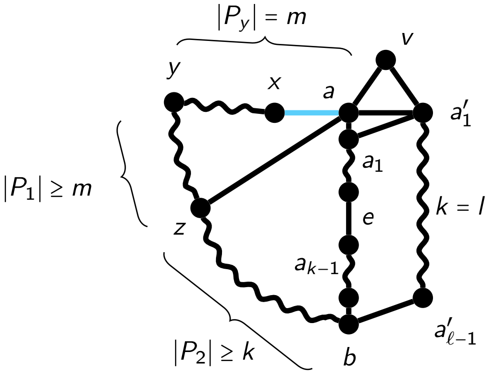
# Monitoring in the General Case




## Theorem [FMMSST24]

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

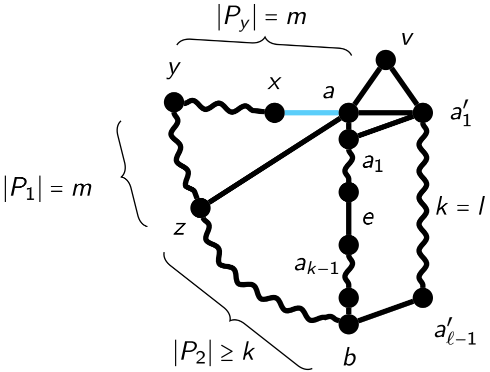
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

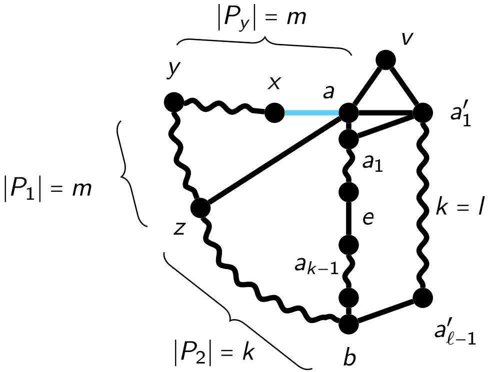
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

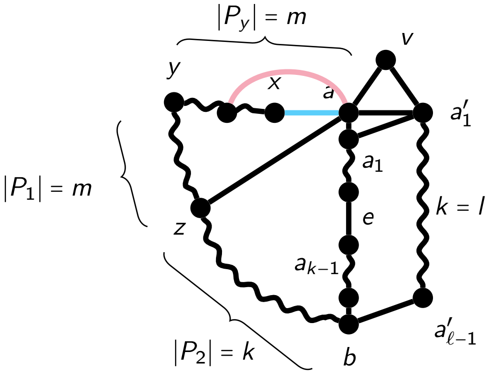
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1 a x$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1 a x z$ is a cycle.

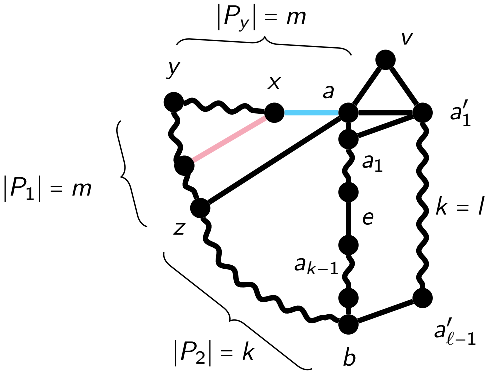
# Monitoring in the General Case




### Theorem [FMMSST24]

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

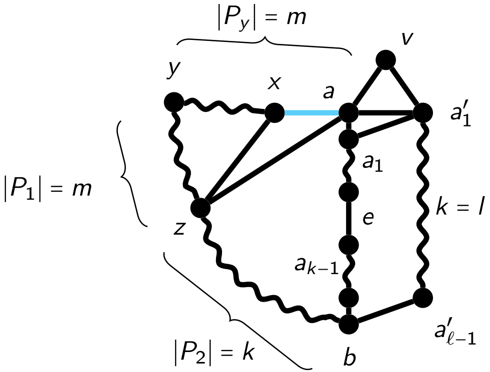
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

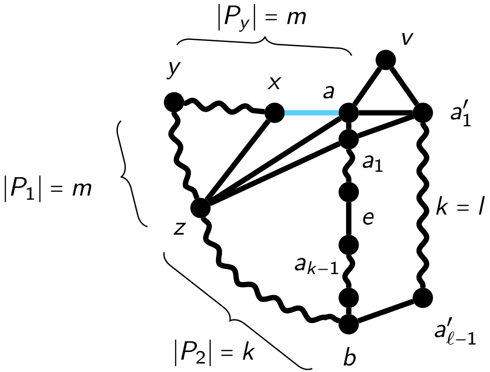
# Monitoring in the General Case




**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

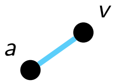
# Monitoring in the General Case



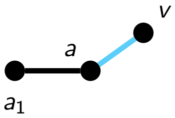

**Theorem [FMMSST24]**

A vertex $a$ is *mandatory* if there exists $a_1$ such that for any $x \in N(u)$ such that $a_1ax$ is an induced 2-path, there exists $z \in V(G)$ such that $a_1axz$ is a cycle.

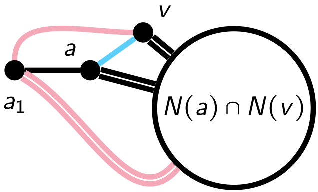
# Monitoring the Edges Incident to $v$



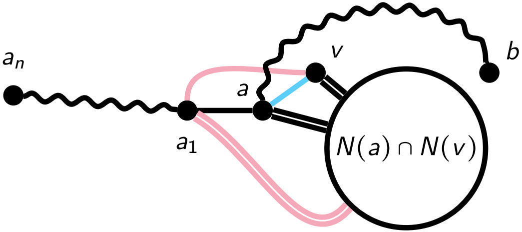
# Monitoring the Edges Incident to $v$



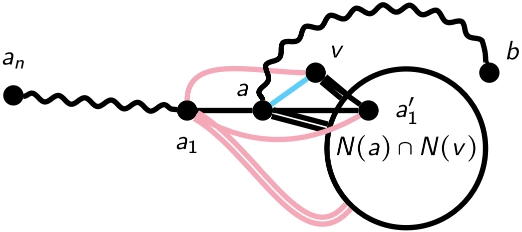
# Monitoring the Edges Incident to $v$



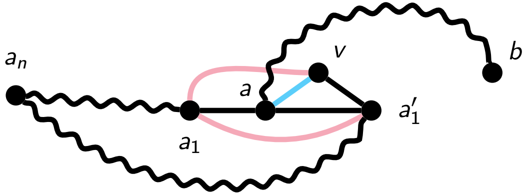
# Monitoring the Edges Incident to $v$



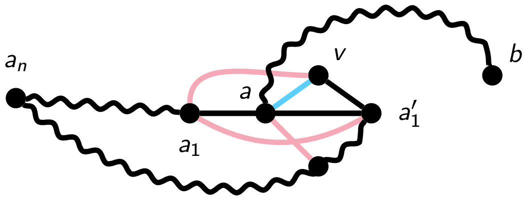
# Monitoring the Edges Incident to $v$



# Monitoring the Edges Incident to $v$



# Monitoring the Edges Incident to $v$



# Conclusion and Future Works

In this work:

- we compute MEG-sets of chordal graphs in polynomial time;
- and we improve the best known algorithm to compute mandatory vertices.

# Conclusion and Future Works

In this work:

- we compute MEG-sets of chordal graphs in polynomial time;
- and we improve the best known algorithm to compute mandatory vertices.

What this result opens:

- Finding more MEG-minimal classes of graphs (all the way up to Perfect);
- investigate polynomial algorithms for non-MEG-minimal classes;
- new intuitions for parameterised algorithms.