Recontamination helps a lot to hunt a rabbit

Thomas Dissaux [1] Foivos Fioravantes [2] Harmender Galhawat [3]
Nicolas Nisse [1]

[1] Université Côte d'Azur, Inria, CNRS, I3S, France

[2] Department of Theoretical Computer Science, FIT, Czech Technical University in Prague, Czechia

[3] Ben-Gurion University of the Negev, Beersheba, Israel

GRASTA 2023

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Graphs
Games
Optimization
Algorithms
Theoretical
Computer Science

Lets play

Hunters and Rabbit

Graph G, k hunters and one invisible rabbit. The rabbit goes on an initial vertex. Then, at each round:

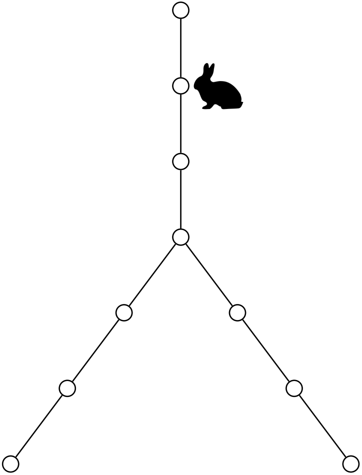
- The hunters **shoot** k vertices of G;
- The rabbit, if not shot, **must** move to an adjacent vertex.

The hunters win iff the rabbit is shot at some round.

Definition

The **hunter number** of G, denoted $h(G)$, is the minimum number of hunters needed to win.

Example 1 - Remember: rabbit is **invisible**

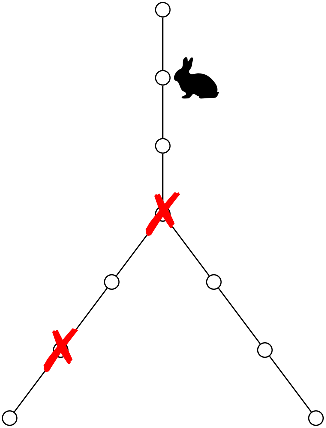


Round 0

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

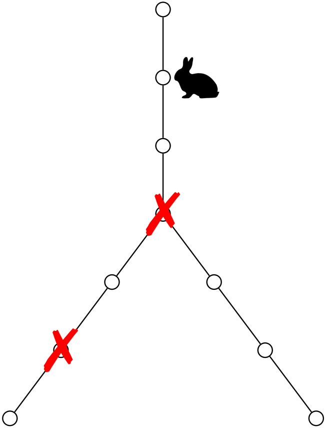


Round $1a$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

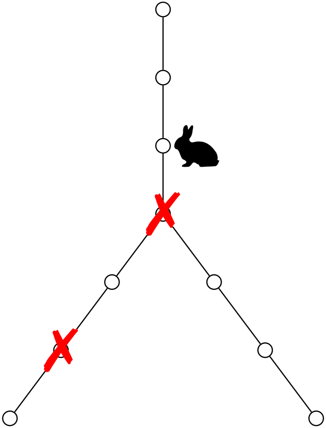


Round $1a$ Round $1b$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

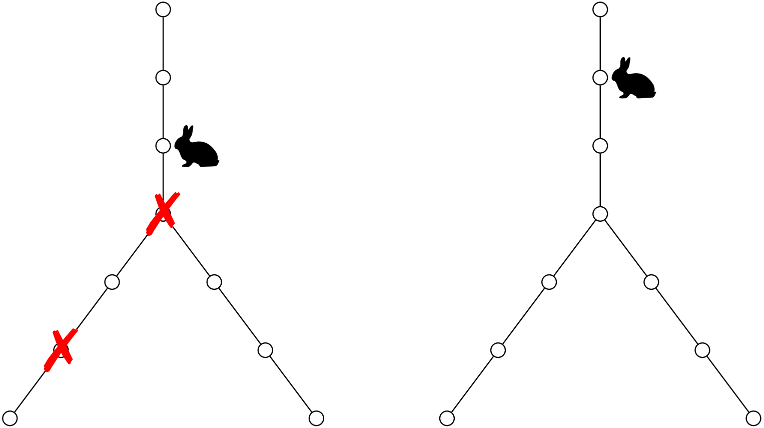


Round $2a$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

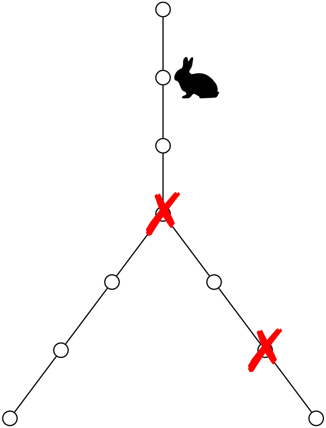


Round $2a$ Round $2b$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

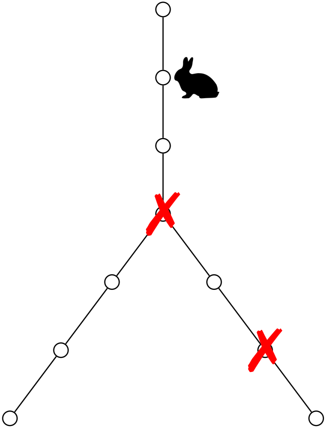


Round $3a$

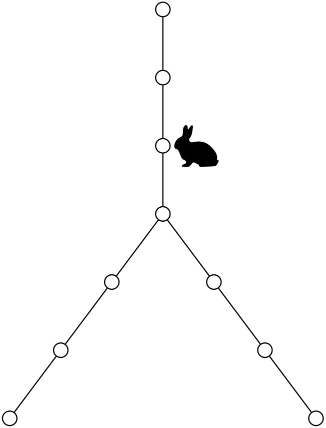
Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**



Round $3a$

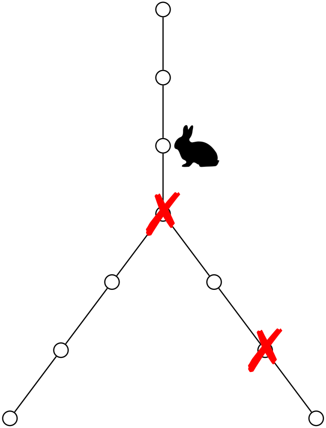


Round $3b$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**

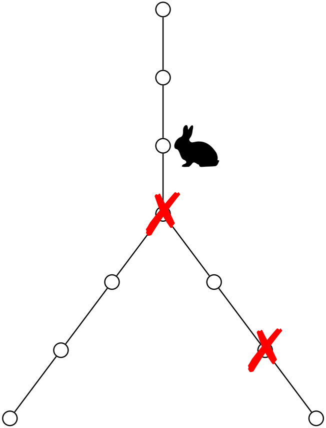


Round $4a$

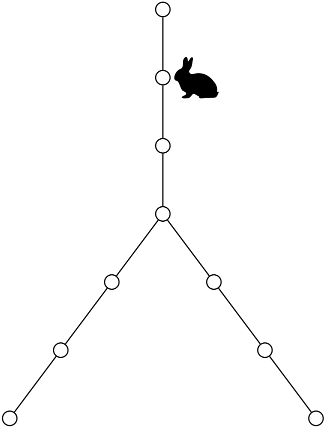
Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**



Round $4a$

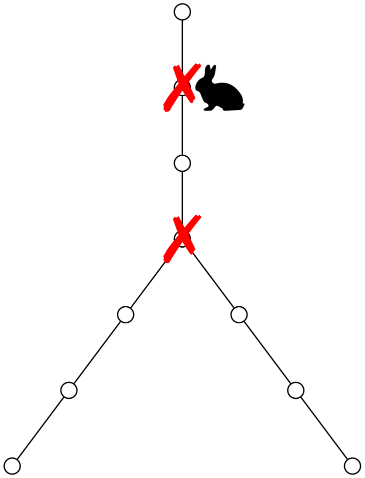


Round $4b$

Remember

The rabbit is **invisible** and **must move** in every round.

Example 1 - Remember: rabbit is **invisible**



Round 5

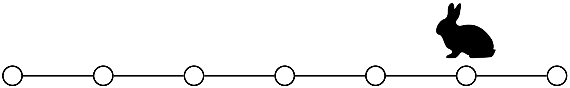
Observation

- The area that is available to the rabbit **does not increase** ↑ **monotonicity property**
- A **strategy** for 2 hunters to win
- This graph has **hunter number** $h(G) \leq 2$
- Smallest tree with $h(G) = 2$ (2013, Britnell and Wildon)

Attention!

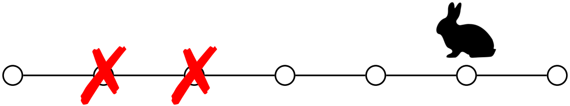
It was **not** necessary to shoot on all vertices of G.

Example 2



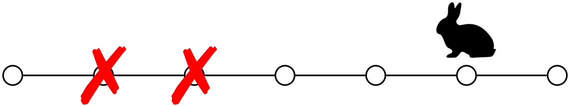
Round 0

Example 2

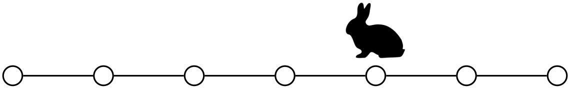


Round $1a$

Example 2

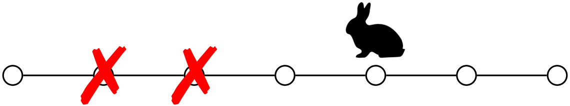


Round 1a



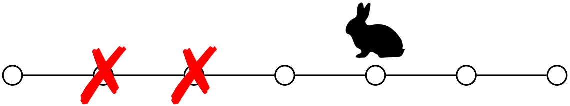
Round 1b

Example 2

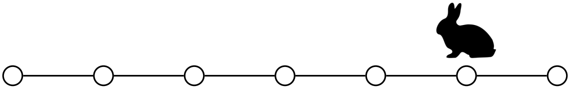


Round $2a$

Example 2

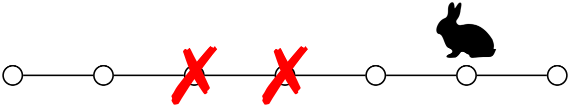


Round $2a$



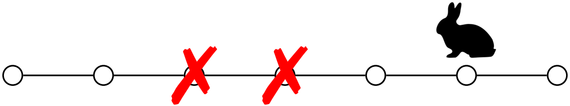
Round $2b$

Example 2

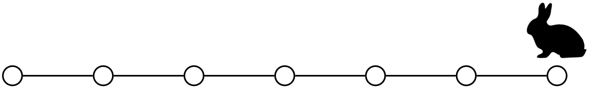


Round $3a$

Example 2

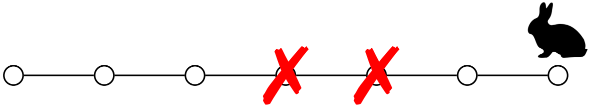


Round $3a$



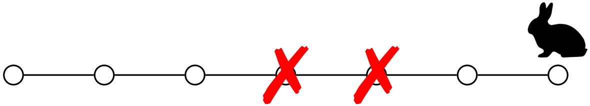
Round $3b$

Example 2

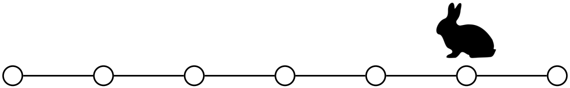


Round $4a$

Example 2

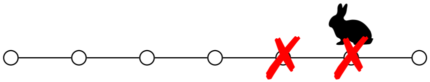


Round $4a$



Round $4b$

Example 2



Round 5

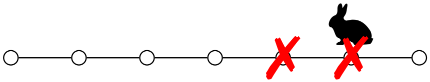
Observation

- The area that is available to the rabbit **does not increase** ↑ **monotonicity property**
- A **strategy** for 2 hunters to win
- This graph has **hunter number** $h(G) \leq 2$

Attention!

It was **not** necessary to shoot on all vertices of G.

Example 2



Round 5

Observation

- The area that is available to the rabbit **does not increase** ↑ **monotonicity property**
- A **strategy** for 2 hunters to win
- This graph has **hunter number** $h(G) \leq 2$

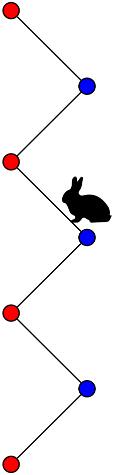
Attention!

It was **not** necessary to shoot on all vertices of G.

Is this optimal? Strategy with only ONE hunter?

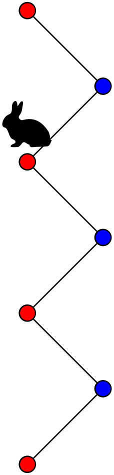
Bipartite graphs are weird

Rabbit starts on **blue** vertex



Round 0

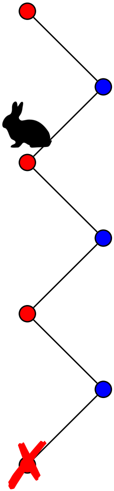
Rabbit starts on **red** vertex



Round 0

Bipartite graphs are weird

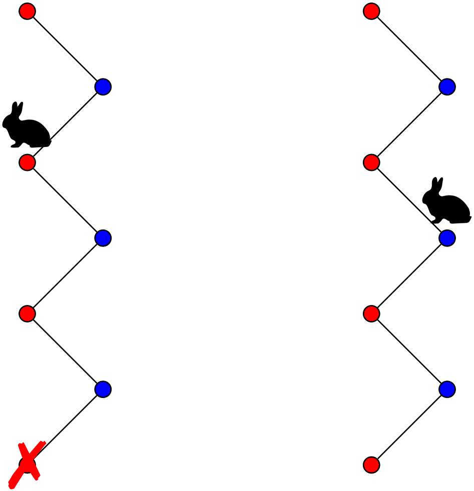
Rabbit starts on **red** - Hunter starts on **red**



Round $1a$

Bipartite graphs are weird

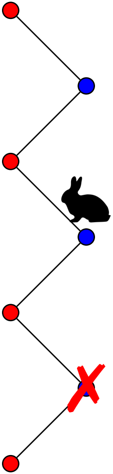
Rabbit starts on **red** - Hunter starts on **red**



Round 1*a* Round 1*b*

Bipartite graphs are weird

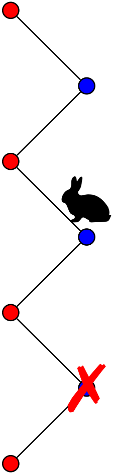
Rabbit starts on **red** - Hunter starts on **red**



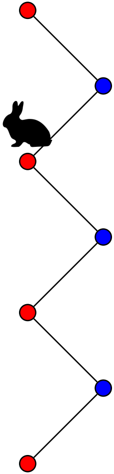
Round $2a$

Bipartite graphs are weird

Rabbit starts on **red** - Hunter starts on **red**



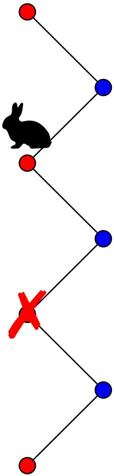
Round $2a$



Round $2b$

Bipartite graphs are weird

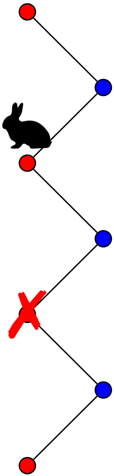
Rabbit starts on **red** - Hunter starts on **red**



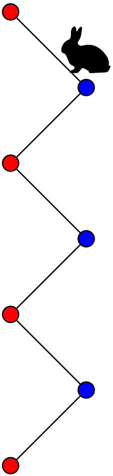
Round $3a$

Bipartite graphs are weird

Rabbit starts on **red** - Hunter starts on **red**



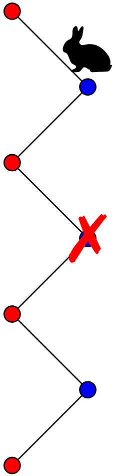
Round $3a$



Round $3b$

Bipartite graphs are weird

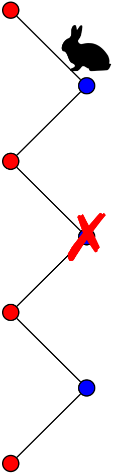
Rabbit starts on **red** - Hunter starts on **red**



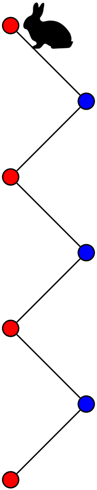
Round $4a$

Bipartite graphs are weird

Rabbit starts on **red** - Hunter starts on **red**



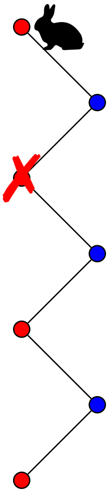
Round $4a$



Round $4b$

Bipartite graphs are weird

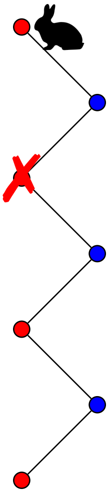
Rabbit starts on **red** - Hunter starts on **red**



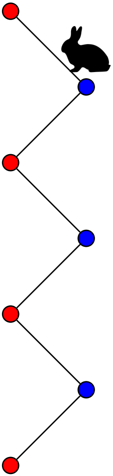
Round $5a$

Bipartite graphs are weird

Rabbit starts on **red** - Hunter starts on **red**



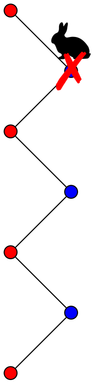
Round $5a$



Round $5b$

Bipartite graphs are weird

Rabbit starts on **red** - Hunter starts on **red**



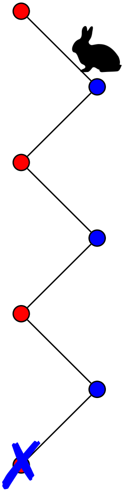
Round 6

Observation

- Rabbit switches colour every round
- Hunter shoots **consecutively** one by one all the vertices
- Hunter shoots same colour as the one occupied by the rabbit in each round
- The area that is available to the rabbit **does not increase**
  ↑ **monotonicity property**
- A **strategy** for 1 hunter to win **if** hunter and rabbit start on **same colour**
- What if hunter and rabbit start on **different colours**?

Bipartite graphs are very weird

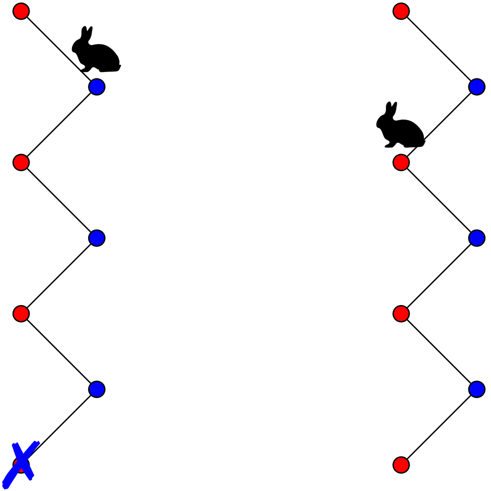
Rabbit starts on **blue** - Hunter starts on **red**



Round $1a$

Bipartite graphs are very weird

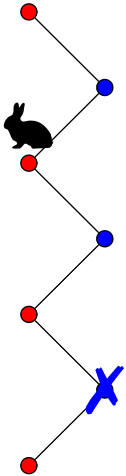
Rabbit starts on **blue** - Hunter starts on **red**



Round 1a Round 1b

Bipartite graphs are very weird

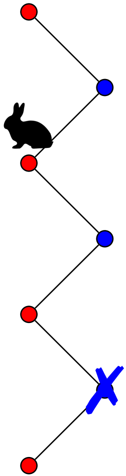
Rabbit starts on **blue** - Hunter starts on **red**



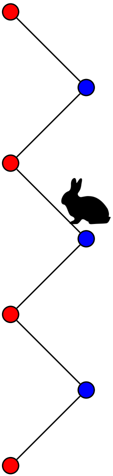
Round $2a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



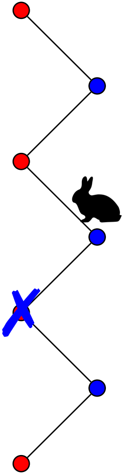
Round $2a$



Round $2b$

Bipartite graphs are very weird

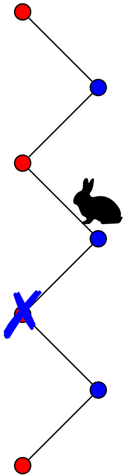
Rabbit starts on **blue** - Hunter starts on **red**



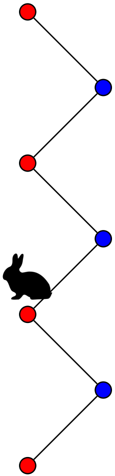
Round $3a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



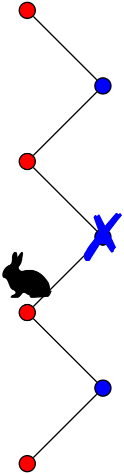
Round $3a$



Round $3b$

Bipartite graphs are very weird

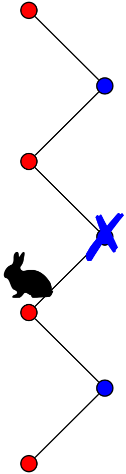
Rabbit starts on **blue** - Hunter starts on **red**



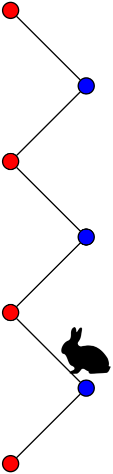
Round $4a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



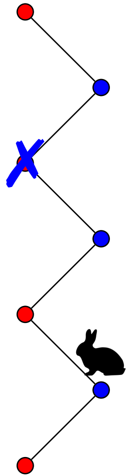
Round $4a$



Round $4b$

Bipartite graphs are very weird

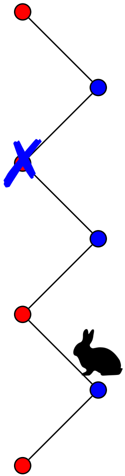
Rabbit starts on **blue** - Hunter starts on **red**



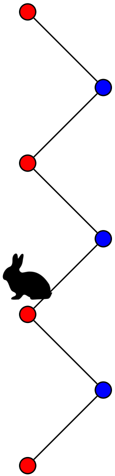
Round $5a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



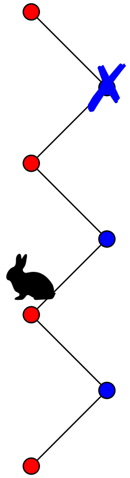
Round $5a$



Round $5b$

Bipartite graphs are very weird

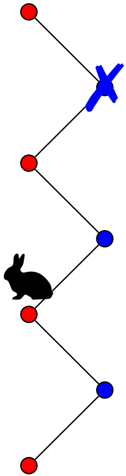
Rabbit starts on **blue** - Hunter starts on **red**



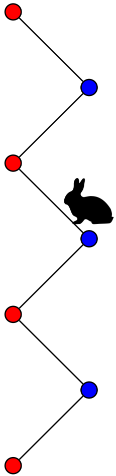
Round $6a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



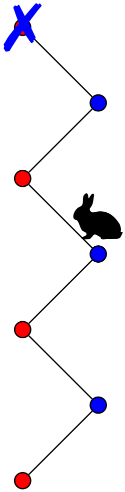
Round $6a$



Round $6b$

Bipartite graphs are very weird

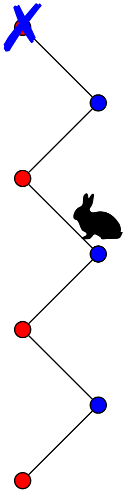
Rabbit starts on **blue** - Hunter starts on **red**



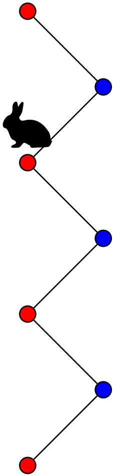
Round $7a$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



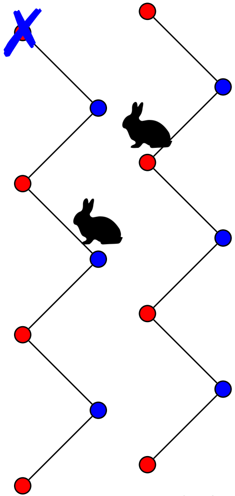
Round $7a$



Round $7b$

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



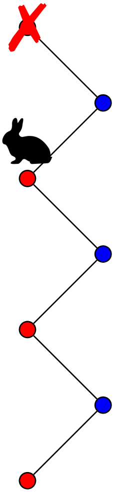
Round $7a$

Round $7b$

- If rabbit still alive ⇒
- Hunter started from wrong colour
- But only two colours

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**

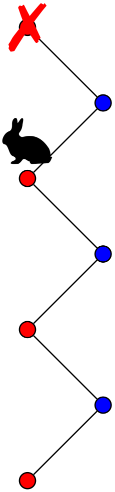


Round $8a$

- If rabbit still alive ⇒
- Hunter started from wrong colour
- But only two colours
- Hunter switches colour

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



Round $8a$

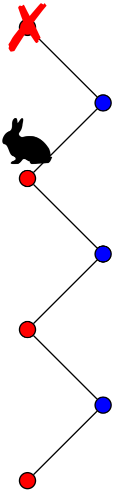
- If rabbit still alive ⇒
- Hunter started from wrong colour
- But only two colours
- Hunter switches colour
- Now Hunter shoots same colour as the one occupied by the rabbit
- Same as before

Bipartite Lemma (2016, Abramovskaya et al.)

In bipartite graphs, assume we know the starting colour of the rabbit.

Bipartite graphs are very weird

Rabbit starts on **blue** - Hunter starts on **red**



Round $8a$

- If rabbit still alive $\Rightarrow$
- Hunter started from wrong colour
- But only two colours
- Hunter switches colour
- Now Hunter shoots same colour as the one occupied by the rabbit
- Same as before

Bipartite Lemma (2016, Abramovskaya et al.)

In bipartite graphs, assume we know the starting colour of the rabbit.

Attention! During first "pass" of the path, the area available to the rabbit was **unaffected** $\Rightarrow$ Not monotone (in the classical sense)!

What is known already

Finding a princess in a palace (2013, Britnell and Wildon)

- Introduced the problem for **one** hunter
- Any tree T has $h(T) = 1$ if and only if it does **not** contain the tree of the example as a subgraph.
- Particular behaviour of paths

Hunters and Rabbit (2016, Abramovskaya et al.)

- Generalised for many hunters
- Precise values for cycles, complete graphs, grids, hypercubes
- Particular behaviour of bipartite graphs + first upper bound for trees

Catching a mouse on a tree (2015, Gruslys and Meroueh)

- For any tree T, we have $h(T) \leq \lceil \frac{1}{2} \log_2(|V(T)|) \rceil$

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.

[1] monotone = "the area available to the rabbit does not increase"

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.
- Introduced the **monotone bipartite hunter number** $\boldsymbol{mh_B(G)}$.

[1] monotone = "the area available to the rabbit does not increase"

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.
- Introduced the **monotone bipartite hunter number** $\boldsymbol{mh_B(G)}$.

Relation with the pathwidth

For any graph G,
$pw(G) \leq mh(G) \leq pw(G) + 1$.

[1] monotone = "the area available to the rabbit does not increase"

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.
- Introduced the **monotone bipartite hunter number** $\boldsymbol{mh_B(G)}$.

Relation with the **pathwidth**

For any graph G,
$pw(G) \leq mh(G) \leq pw(G) + 1$.

Compute $mh(G)$:

- split and interval graphs
- cographs
- trees

[1] monotone = "the area available to the rabbit does not increase"

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.
- Introduced the **monotone bipartite hunter number** $\boldsymbol{mh_B(G)}$.

Relation with the **pathwidth**

For any graph G,
$pw(G) \leq mh(G) \leq pw(G) + 1$.

Compute $mh(G)$:

- split and interval graphs
- cographs
- trees

Recontamination helps a lot

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

[1] monotone = "the area available to the rabbit does not increase"

Our results

- Introduced the **monotone**[1] **hunter number** $\boldsymbol{mh(G)}$.
- Introduced the **monotone bipartite hunter number** $\boldsymbol{mh_B(G)}$.

Relation with the **pathwidth**

For any graph G,
$pw(G) \leq mh(G) \leq pw(G) + 1$.

Compute $mh(G)$:

- split and interval graphs
- cographs
- trees

Recontamination helps a lot

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

General positive result

Deciding if $h(G) \leq k$ is in FPT when parameterised by the vertex cover number of G.

[1] monotone = "the area available to the rabbit does not increase"

Monotonicity
Pathwidth

Why monotonicity?

Recall: monotonicity = "the area available to the rabbit does not increase"
Classical notion in Graph Searching because:

- easier to design monotone strategies
- take time polynomial to the size of the input

Also, monotonicity links Graph Searching and:

- pathwidth (1991, Bienstock and Seymour)
- treewidth (1993, Seymour and Thomas)

and is fundamental behind:

Theorem (1994, Ellis, Sudborough, and Turner)

Polynomial algorithm to compute the pathwidth of a tree.

Why monotonicity?

Recall: monotonicity = "the area available to the rabbit does not increase"
Classical notion in Graph Searching because:

- easier to design monotone strategies
- take time polynomial to the size of the input

Also, monotonicity links Graph Searching and:

- pathwidth (1991, Bienstock and Seymour)
- treewidth (1993, Seymour and Thomas)

and is fundamental behind:

Theorem (1994, Ellis, Sudborough, and Turner)

Polynomial algorithm to compute the pathwidth of a tree.

But

Classical monotonicity **fails** for our problem.

A particular version of monotonicity

A vertex $\boldsymbol{v}$ is cleared[a] at round $\boldsymbol{i}$ if either:

[a] The rabbit is no longer supposed to be here.

- v is shot at round i or
- Neighbours of v that could host the rabbit are shot at round i

A particular version of monotonicity

A vertex v is **cleared**[a] **at round** i if either:

[a] The rabbit is no longer supposed to be here.

- v is shot at round i or
- Neighbours of v that could host the rabbit are shot at round i

Monotone strategy $\rightarrow mh(G)$

A **monotone** strategy guarantees that if the rabbit goes on a **cleared** vertex, it is shot immediately.

Monotone bipartite strategy $\rightarrow mh_B(G)$

Same as monotone + assume knowledge of initial colour of the rabbit.

A particular version of monotonicity

A vertex $\boldsymbol{v}$ is **cleared**[a] **at round** $\boldsymbol{i}$ if either:

[a] The rabbit is no longer supposed to be here.

- v is shot at round i or
- Neighbours of v that could host the rabbit are shot at round i

Monotone strategy $\rightarrow mh(G)$

A **monotone** strategy guarantees that if the rabbit goes on a **cleared** vertex, it is shot immediately.

Monotone bipartite strategy $\rightarrow mh_B(G)$

Same as monotone + assume knowledge of initial colour of the rabbit.

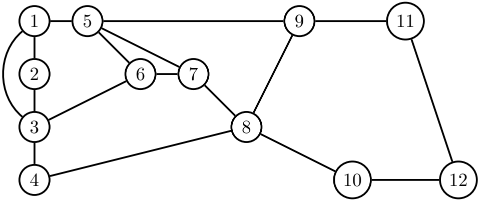
Observe

For bipartite graphs, $mh_B(G) \leq mh(G)$.

Example

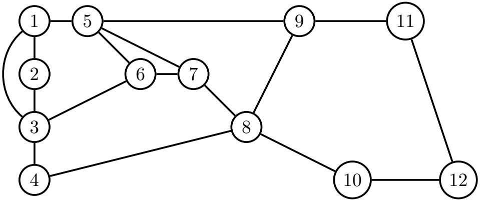
For $n \geq 4$, $mh(P_n) = 2$ but $mh_B(P_n) = 1$.

Pathwidth - definition



G

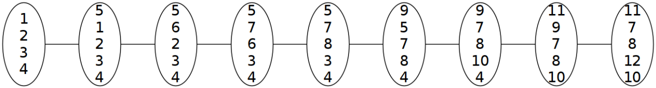
Pathwidth - definition



G

Decompose graph into **bags**:

1. all vertices appear in some bags
2. all edges appear in some bags
3. bags sharing a vertex form a path



A path decomposition of G

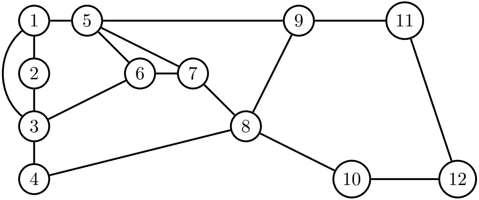
Pathwidth $pw(G)$ = size of largest bag -1. Here, $pw(G) \leq 4$.

Monotone hunter number and pathwidth

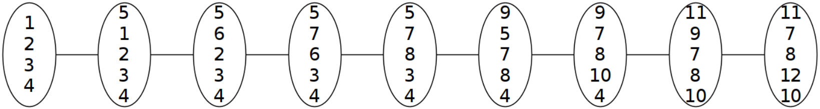
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

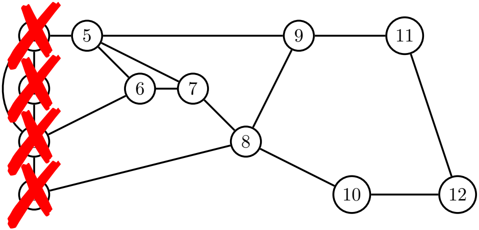


Monotone hunter number and pathwidth

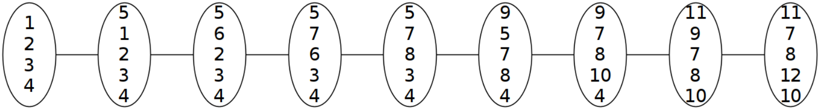
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

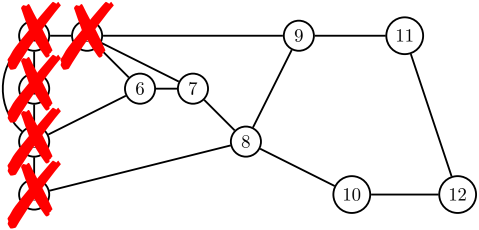


Monotone hunter number and pathwidth

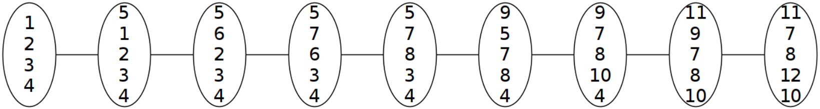
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



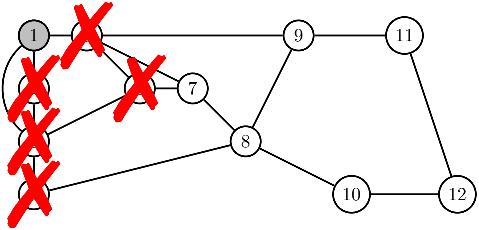
Gray = cleared



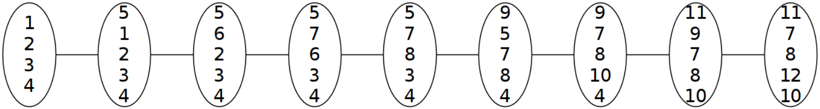
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow$ $\boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

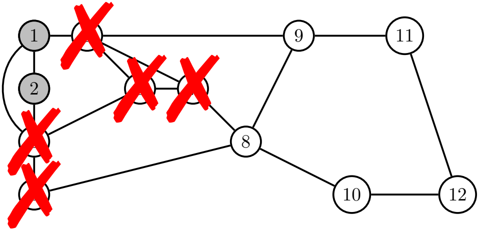


Monotone hunter number and pathwidth

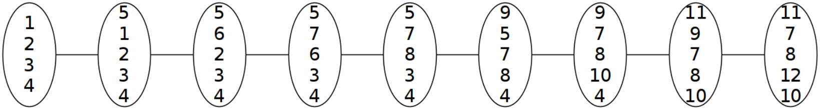
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

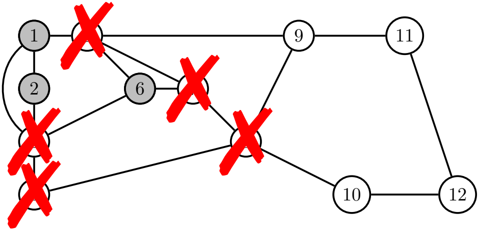


Monotone hunter number and pathwidth

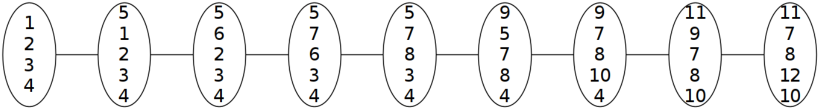
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

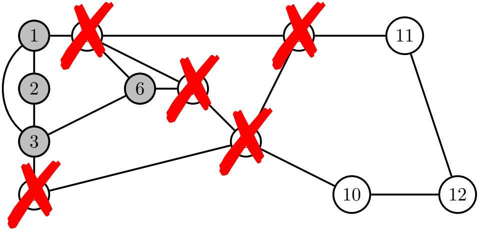


Monotone hunter number and pathwidth

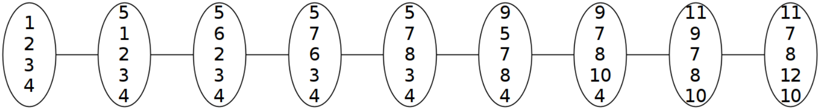
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

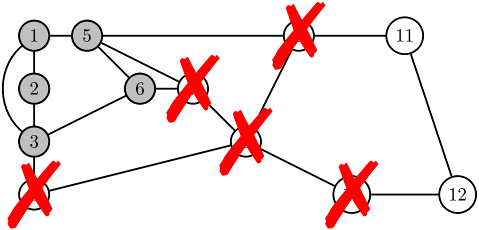


Monotone hunter number and pathwidth

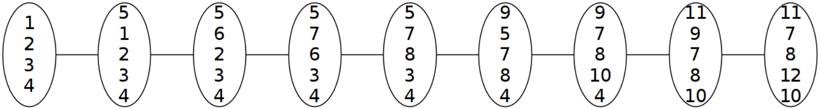
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



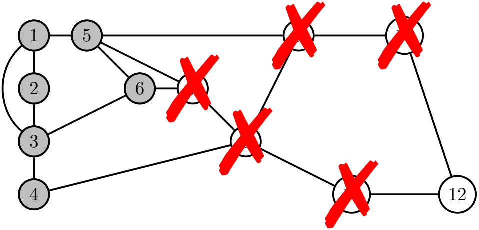
Gray = cleared



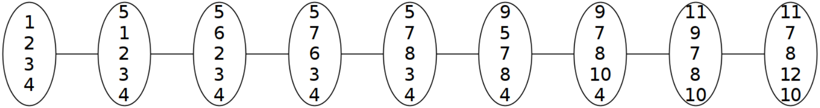
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



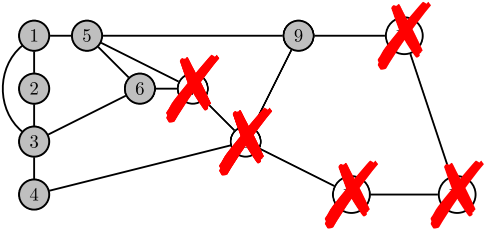
Gray = cleared



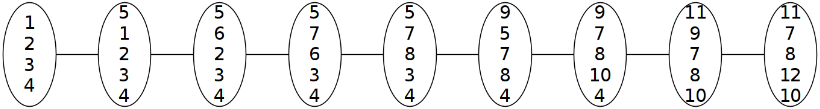
Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \leq pw(G) + 1}$: Shoot vertices according to the path decomposition. Already observed in (2016, Abramovskaya et al.).



Gray = cleared

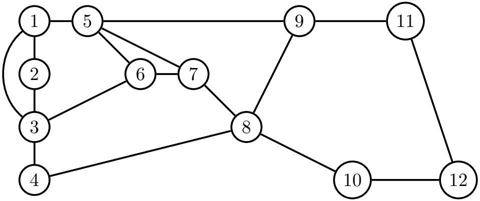


Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

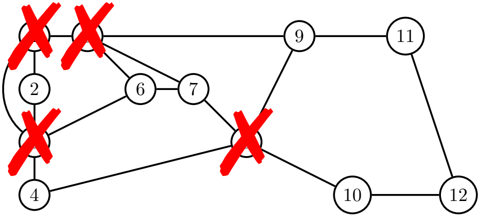
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

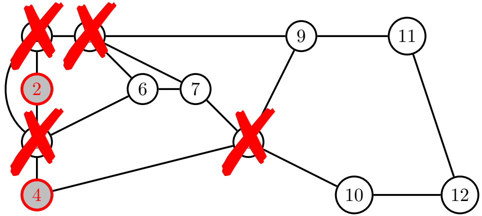
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

$S_1 = \{1, 3, 5, 8\}$

$S_2 = \{1, 3, 5, 8\}$

$S_3 = \{3, 5, 6, 8\}$

$S_4 = \{5, 6, 8, 9\}$

$S_5 = \{8, 9, 10, 11\}$

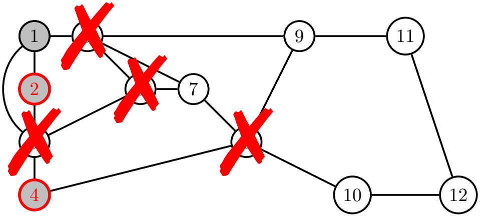
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

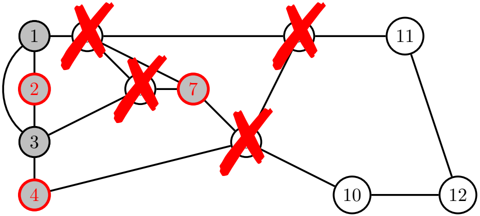
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

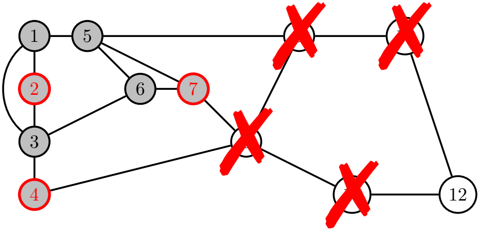
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

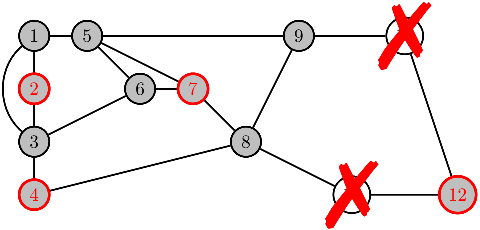
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

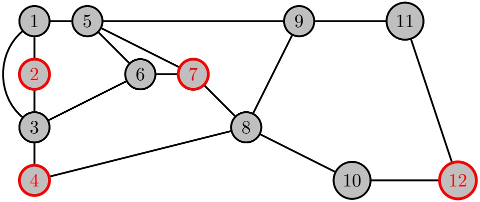
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

$S_1 = \{1, 3, 5, 8\}$

$S_2 = \{1, 3, 5, 8\}$

$S_3 = \{3, 5, 6, 8\}$

$S_4 = \{5, 6, 8, 9\}$

$S_5 = \{8, 9, 10, 11\}$

$S_6 = \{10, 11\}$

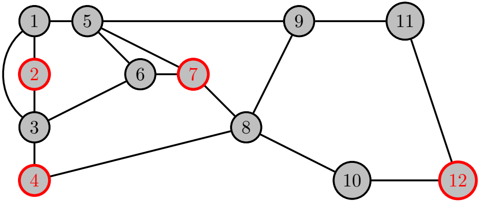
Sequence of shots, **almost** a path decomposition.

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:

$S_1 = \{1, 3, 5, 8\}$

$S_2 = \{1, 3, 5, 8\}$

$S_3 = \{3, 5, 6, 8\}$

$S_4 = \{5, 6, 8, 9\}$

$S_5 = \{8, 9, 10, 11\}$

$S_6 = \{10, 11\}$

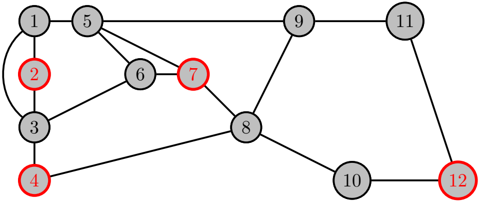
Sequence of shots, **almost** a path decomposition. Problem with vertices that are cleared whithout being shot.

Monotone hunter number and pathwidth

Theorem

For any graph G, $pw(G) \leq mh(G) \leq pw(G) + 1$.

$\rightarrow \boldsymbol{mh(G) \geq pw(G)}$:



Shots:
$S_1 = \{1, 3, 5, 8\}$
$S_2 = \{1, 3, 5, 8\}$
$S_3 = \{3, 5, 6, 8\}$
$S_4 = \{5, 6, 8, 9\}$
$S_5 = \{8, 9, 10, 11\}$
$S_6 = \{10, 11\}$

Sequence of shots, **almost** a path decomposition. Problem with vertices that are cleared whithout being shot.
$\rightarrow$ Create intermediary bags: $S^1_{1,2} = \{1, 2, 3, 5, 8\}$, $S^2_{1,2} = \{1, 3, 4, 5, 8\}$, $S_{3,4} = \{3, 5, 6, 7, 8\}$, $S_{5,6} = \{8, 9, 10, 11, 12\}$.

Recontamination helps a lot

Algorithm to compute $h(T)$?

Classic approach for trees:

- Define the monotone version
- Compute an optimal monotone strategy
- Transform any optimal monotone strategy into a non-monotone with same number of searchers

Algorithm to compute $h(T)$?

Classic approach for trees:

- Define the monotone version ✓
- Compute an optimal monotone strategy ✓
- Transform any optimal monotone strategy into a non-monotone with same number of searchers

Theorem

Polynomial-time algorithm that computes $mh(T)$, for any tree T.

Same approach as (1994, Ellis, Sudborough, and Turner).

Algorithm to compute $h(T)$?

Classic approach for trees:

- Define the monotone version ✓
- Compute an optimal monotone strategy ✓
- Transform any optimal monotone strategy into a non-monotone with same number of searchers ✗

Theorem

Polynomial-time algorithm that computes $mh(T)$, for any tree T.

Same approach as (1994, Ellis, Sudborough, and Turner).

Algorithm to compute $h(T)$?

Classic approach for trees:

- Define the monotone version ✓
- Compute an optimal monotone strategy ✓
- Transform any optimal monotone strategy into a non-monotone with same number of searchers ✗

Theorem

Polynomial-time algorithm that computes $mh(T)$, for any tree T.

Same approach as (1994, Ellis, Sudborough, and Turner).

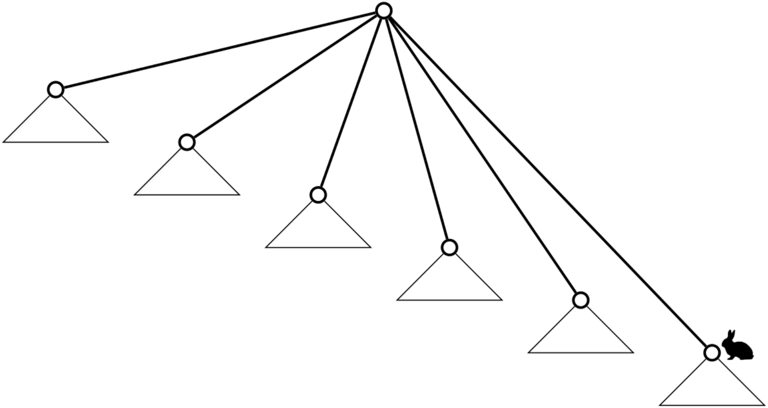
Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

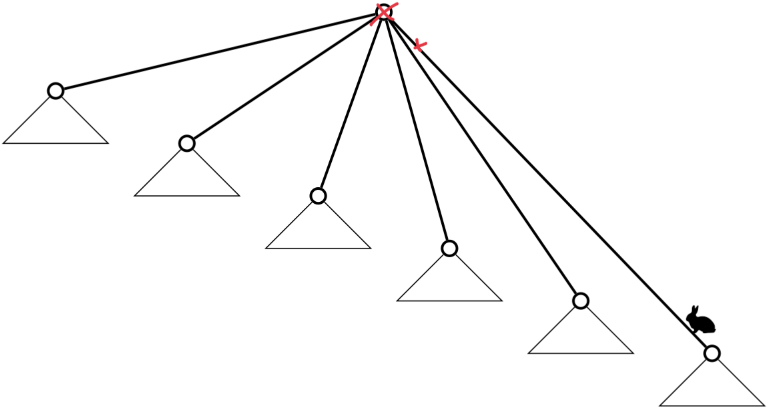


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

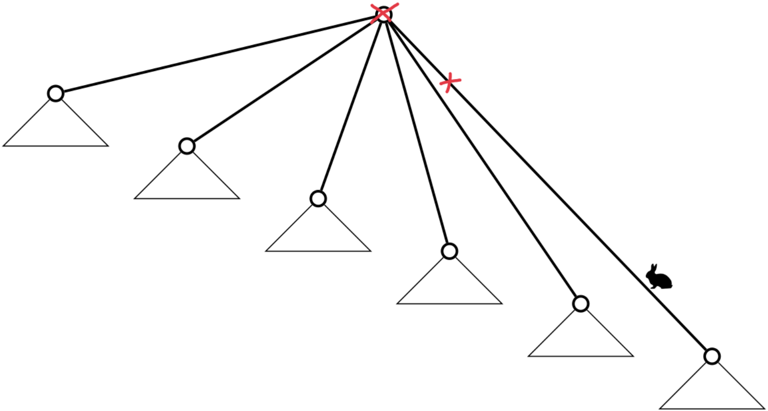


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

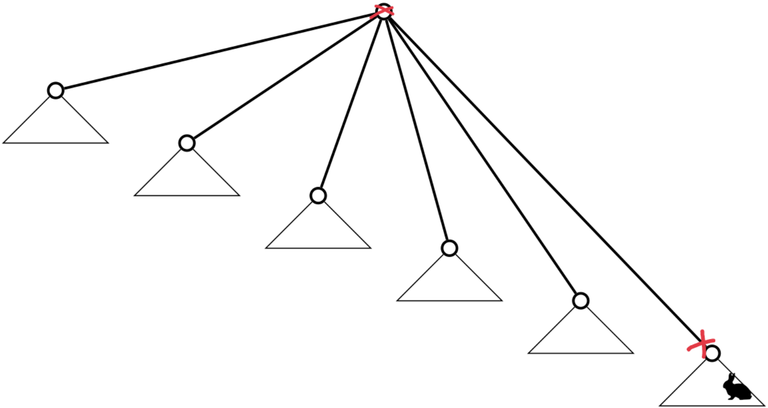


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

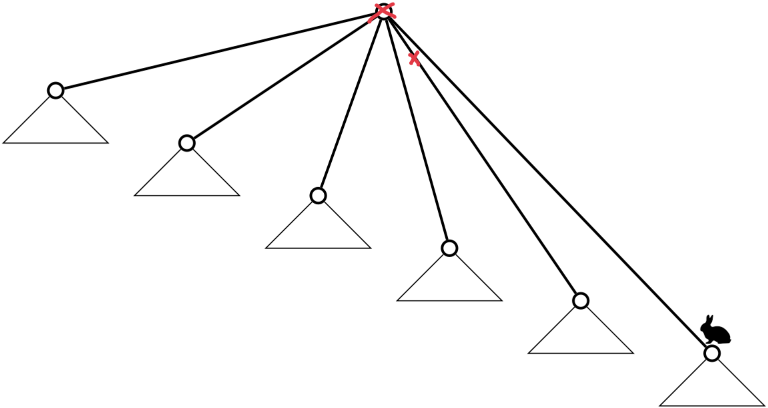


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

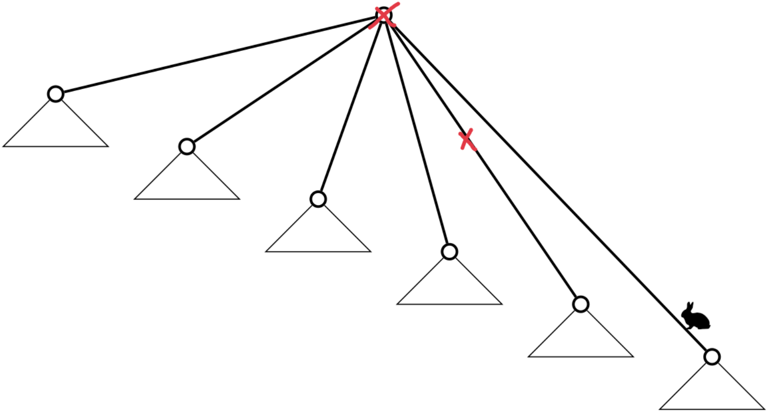


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

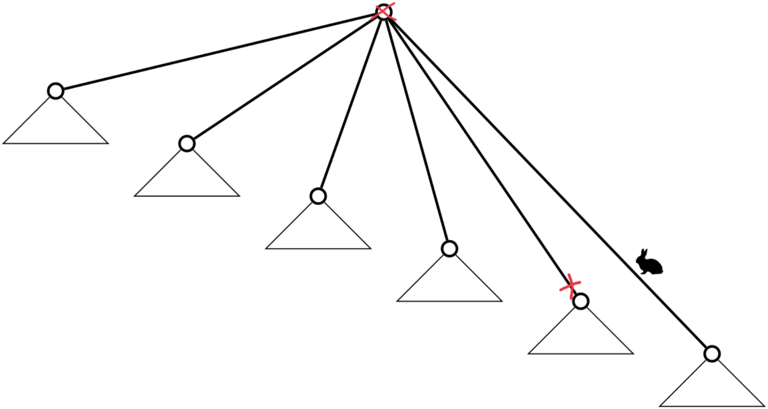


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

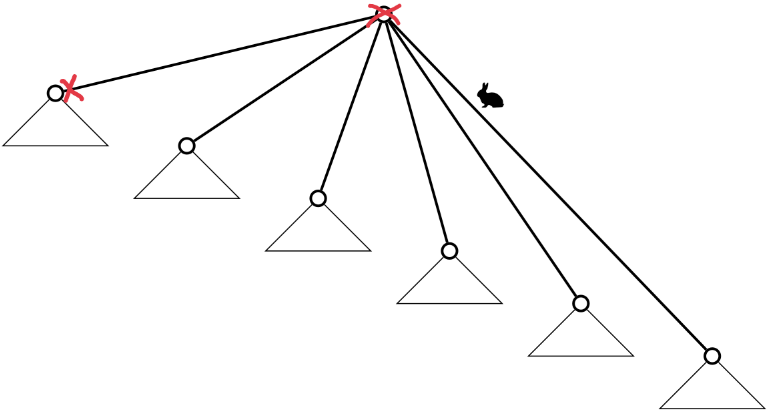


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

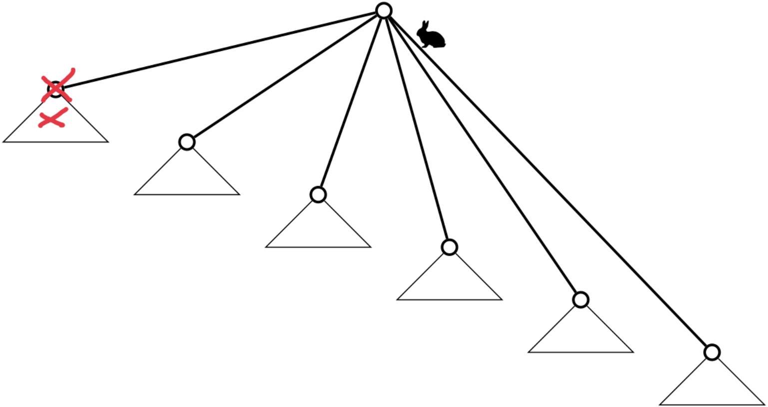


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

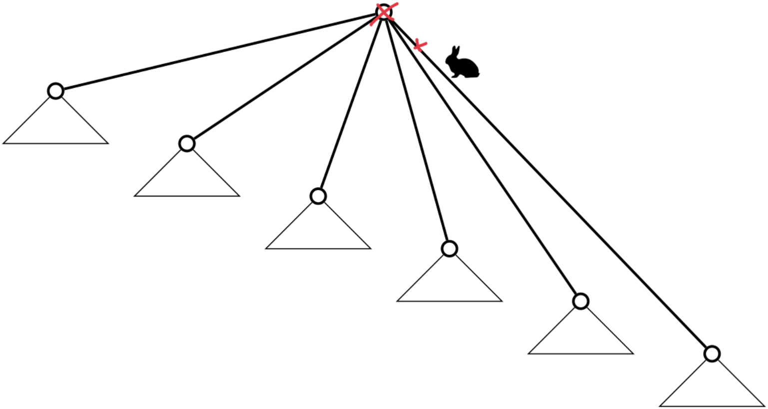


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

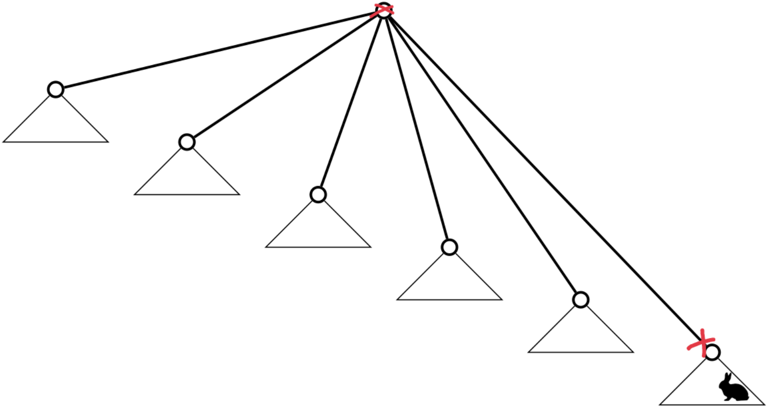


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

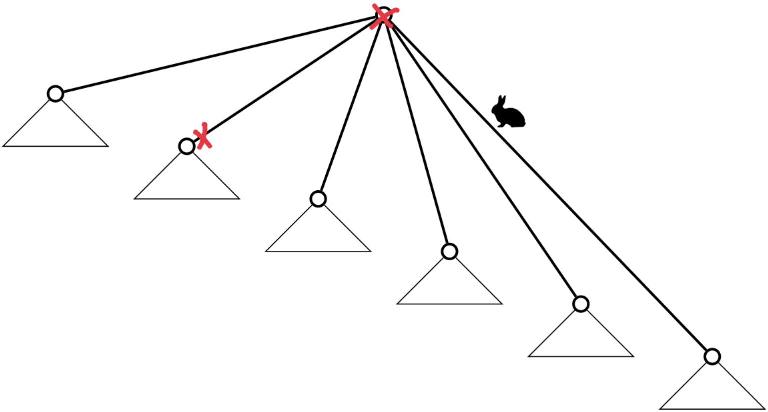


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

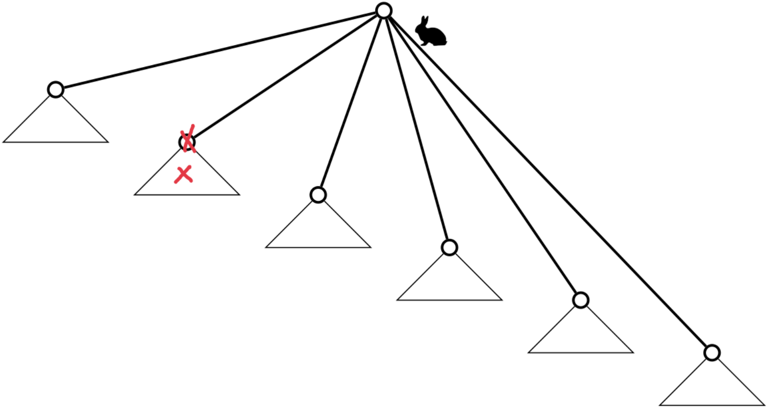


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.

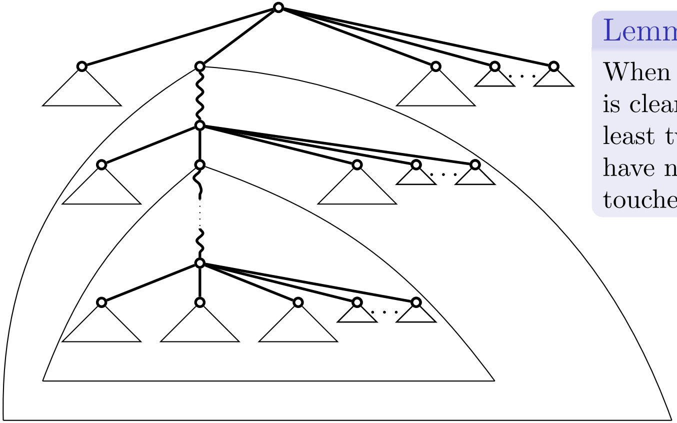


$h(T) = 2$

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.



$mh_B(T) > k$

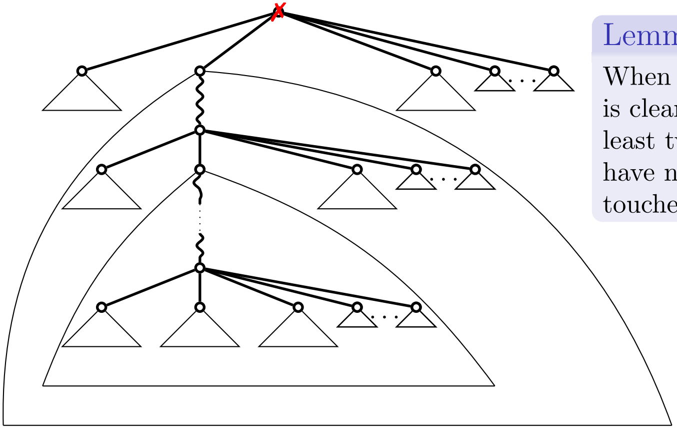
Lemma

When the first branch is cleared, there are at least two branches that have never been touched.

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.



$mh_B(T) > k$

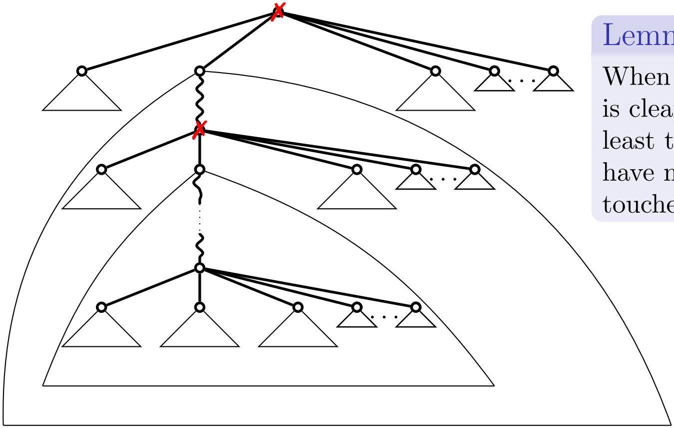
Lemma

When the first branch is cleared, there are at least two branches that have never been touched.

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.



$mh_B(T) > k$

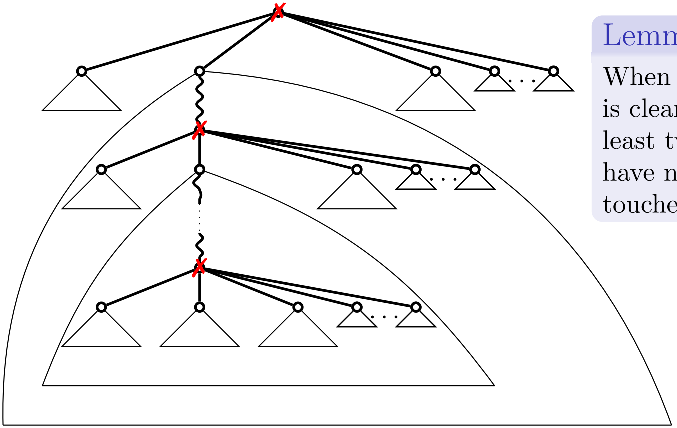
Lemma

When the first branch is cleared, there are at least two branches that have never been touched.

Hunters and Rabbit is not monotone

Theorem

For any k, there exists a tree T such that $h(T) = 2$ and $mh_B(T) \geq k$.



Lemma

When the first branch is cleared, there are at least two branches that have never been touched.

Conclusion

Conclusion

Open questions:

- Polynomial algorithm to compute $h(T)$?
- What is the complexity of computing $h(G)$?
- Is $h(G)$ equivalent to some graphs structural parameter?

Conclusion

Open questions:

- Polynomial algorithm to compute $h(T)$?
- What is the complexity of computing $h(G)$?
- Is $h(G)$ equivalent to some graphs structural parameter?

Grazie!