Algorithmic Complexity Between Structure and Knowledge

How Pursuit-Evasion Games help

Nicolas Nisse

Inria, France

Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, Sophia Antipolis, France

Habilitation à Diriger des Recherches

May 26th 2014

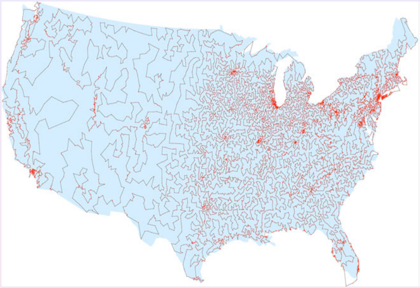
Ph.D. 2007: P. Fraigniaud, LRI, Orsay
Postdoc 2007-08: DIM, Universidad de Chile, Santiago
Postdoc 2008-09: Inria, Mascotte team-project, Sophia Antipolis
since 2009: Chargé de Recherche Inria, COATI team-project

co-PC chair: AlgoTel'13
co-organizer: AlgoTel'11, GRASTA'14
Conference chair: OPODIS'13

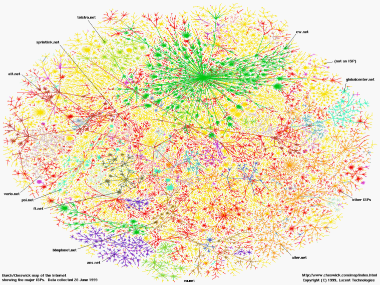
Ph.D. Students: Ronan Soares (November 8th, 2013) and Bi Li (Sept. 2014)

General Context

Finding efficient solutions to problems (routing) arising in telecommunication networks



optimal TSP over 13500 cities [Applegate,Bixby,Chvatal,Cook'98]



Internet 1999 [Cheswicks]

Algorithmic and combinatorial optimization in graphs

Various sources of difficulty

- Problems intrinsically difficult: NP-hard (or more)
- Networks are huge, only partially known, dynamic...

Take Advantage of Structural Properties

Well known: "difficult" problems may become "easy" in particular graph classes

Basic examples where structure helps

- Vertex Cover, Coloring,... in **bipartite** graphs
- Max clique in **interval**/**chordal** graphs
- TSP well approximable in **planar** graphs

Difficulty may arise from the structure

Problem is Fixed Parameter Tractable (FPT) in p [Downey,Fellows'99,Niedermeier'06]

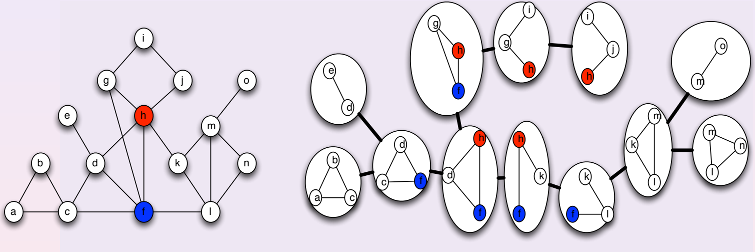
solvable in time $f(k)n^{O(1)}$ in n-node graphs G with parameter $p(G) \leq k$

Decorrelate Complexity and size of the instance
Combinatorial explosion arises from structure not from size

e.g. min. vertex-cover in time $O(2^k \cdot n)$ in n-node graphs with treewidth $\leq k$

Tree/Path-Decompositions

Representation of a graph as a Tree preserving connectivity properties

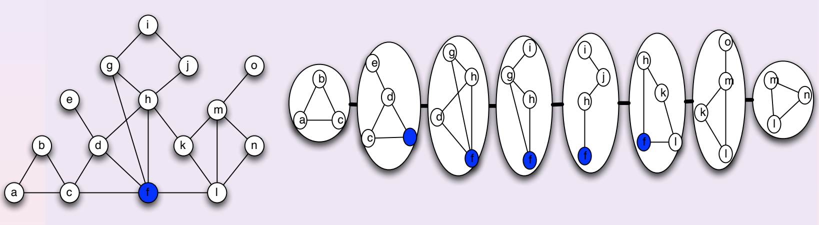


Tree T + family $\mathcal{X}$ of "bags" (set of vertices of G)
Important: intersection of two adjacent bags = separator of G

Width of $(T, \mathcal{X})$: size of largest bag (minus 1)
Treewidth of a graph G, $tw(G)$: min width over all tree-decompositions.

Tree/Path-Decompositions

Representation of a graph as a Path preserving connectivity properties



Path T + family $\mathcal{X}$ of "bags" (set of vertices of G)
Important: intersection of two adjacent bags = separator of G

Width of $(T, \mathcal{X})$: size of largest bag (minus 1)
Pathwidth of a graph G, $pw(G)$: min width over all path-decompositions.

Algorithmic Progress thanks to Treewidth

Dynamic programming on tree/path decomposition

MSOL Problems: "local" problems are FPT in *tw* [Courcelle'90]

e.g., coloring, independent set: $O(2^{tw} n^{O(1)})$; dominating set $O(4^{tw} n^{O(1)})$...

Recent results

Meta-Kernelization (protrusion decomposition) [Bodlaender *et al.*'09]
Single exponential FPT algorithms for "global" properties [Cygan *et al.*'11, Bodlaender *et al.*'13]

Bidimensionality

Subexponential algorithms in H-minor free graphs e.g., [Demaine'08]
based on duality result for treewidth
Graph Minor Theory [Robertson,Seymour 1985-2004]

huge constants may be hidden (at least exponential in *tw*)
"good" decompositions must be computed (computing treewidth is NP-hard)
$\Rightarrow$ How to use/apply in practice?

General Objectives and my Approach

Understand better and actually compute structure to use it for large networks

Understand graph structural properties

New characterizations, new properties..
in general graphs and in real large networks

Compute them

Recognition, efficient computation of properties/decompositions...

Use them

Application on problems in (large) networks (telecommunication, etc.)

Main tool: Pursuit-evasion games

Pursuit-Evasion Games

2-Player games

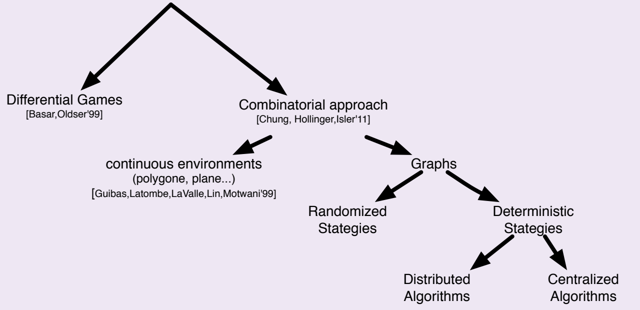
A team of mobile entities (Cops) track down another mobile entity (Robber)

Always one winner

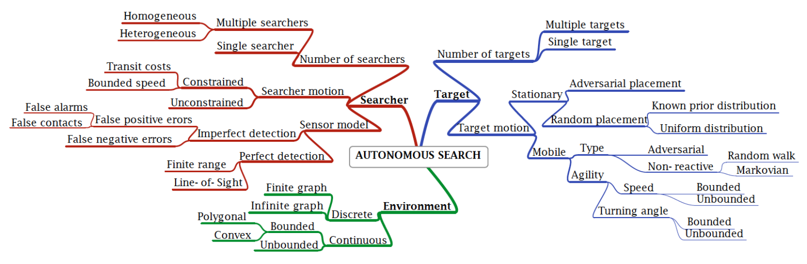
- **Combinatorial Problem:**
  Minimizing some resource for some Player to win
  e.g., minimize number of Cops to capture the Robber.
- **Algorithmic Problem:**
  Computing winning strategy (sequence of moves) for some Player
  e.g., compute strategy for Cops to capture Robber/Robber to avoid the capture

natural applications: coordination of mobile autonomous agents
(Robotic, Network Security, Information Seeking...)
but also: Graph Theory, Models of Computation, Logic, Routing...

Pursuit-Evasion: Over-simplified Classification

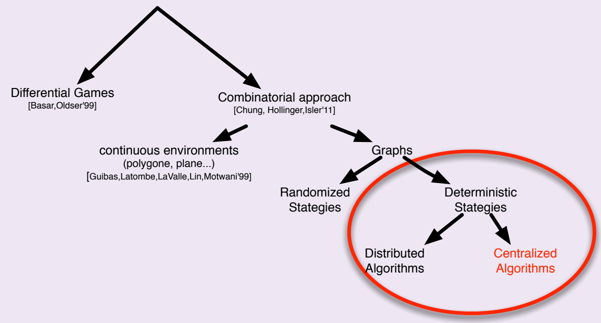


Pursuit-Evasion: Over-simplified Classification



[Chung,Hollinger,Isler'11]

Pursuit-Evasion: Over-simplified Classification



Today: focus on centralized setting

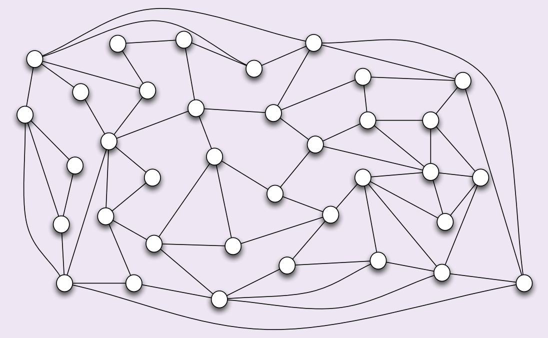
Goal of this talk: illustrate that studying Pursuit-Evasion games helps

- Offer new approaches for several structural graph properties
- Models for studying several practical problems
- Fun and intriguing questions

Outline

Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

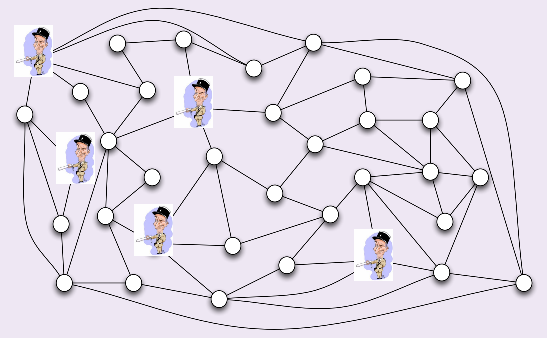
Rules of the $\mathcal{C}\&\mathcal{R}$ game



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

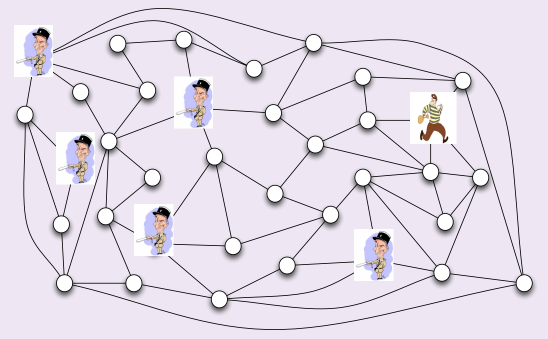
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

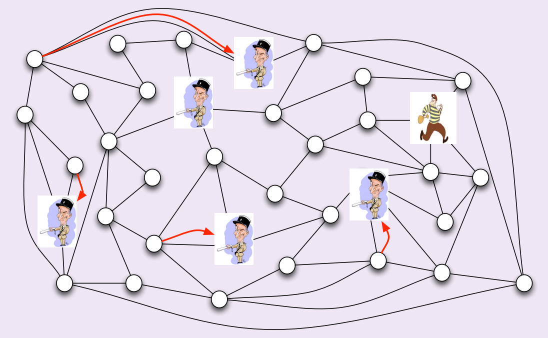
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

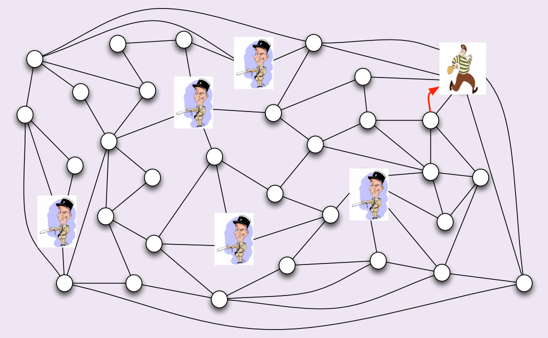
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

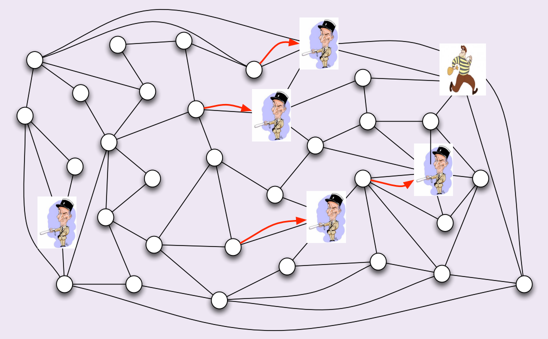
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge



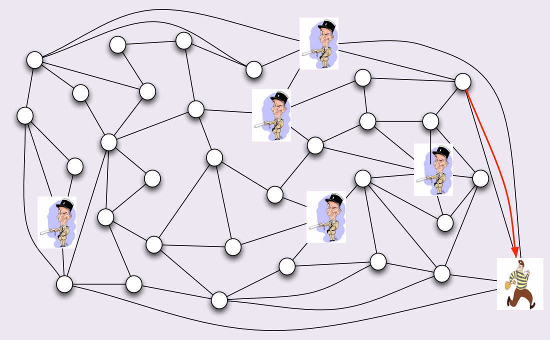
Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops



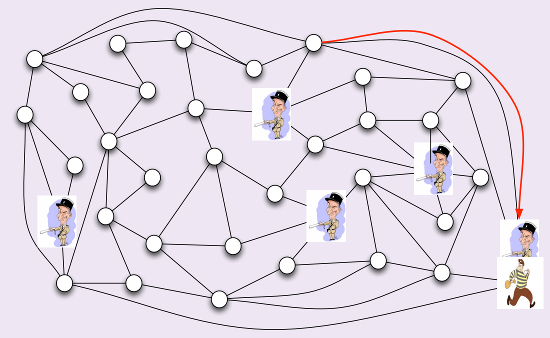
Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops
- Cops must capture Robber (i.e., occupy the same node)



Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

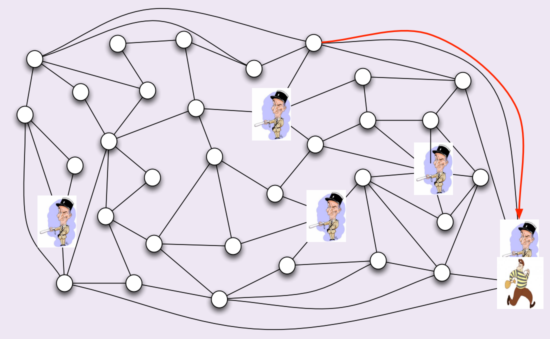
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops
- Cops must capture Robber (i.e., occupy the same node)

Cop Number of a graph G

$cn(G)$: min # Cops to win in G



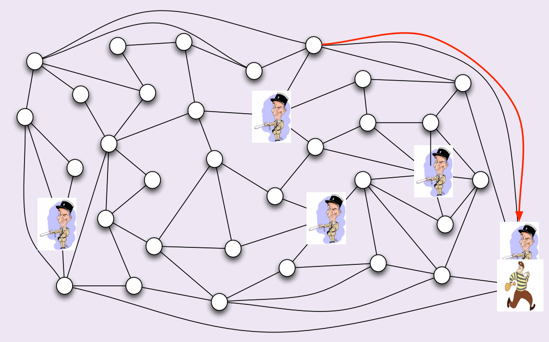
Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops
- Cops must capture Robber (i.e., occupy the same node)



Cop Number of a graph G

$cn(G)$: min # Cops to win in G

Complexity of deciding $cn(G) \leq k$? in general graphs G (a long story)

$n^{O(k)}$-algorithm [BI93], EXPTIME-complete in directed graphs [GR95], NP-hard, W[2] [Fomin,Golovach,Kratochvil,N.,Suchan10], PSPACE-hard [Mamino13], EXPTIME-complete [Kinnersley 14].

Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

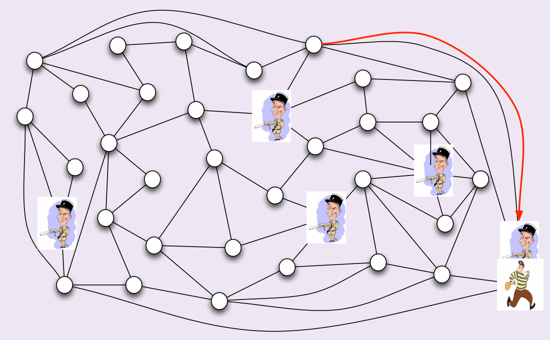
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops
- Cops must capture Robber (i.e., occupy the same node)

Cop Number of a graph G

$cn(G)$: min # Cops to win in G



Link with graph structure? a first surprising (?) example

Cops & Robber Games [Nowakowski and Winkler; Quilliot, 1983]

Rules of the $\mathcal{C}\&\mathcal{R}$ game

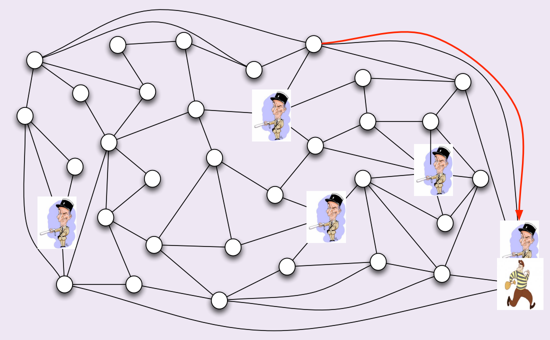
1. Place $k \geq 1$ Cops $\mathcal{C}$ on nodes
2. Visible Robber $\mathcal{R}$ at one node
3. Turn by turn
   (1) each $\mathcal{C}$ slides along ≤ 1 edge
   (2) $\mathcal{R}$ slides along ≤ 1 edge

Goal of the $\mathcal{C}\&\mathcal{R}$ game

- Robber must avoid the Cops
- Cops must capture Robber (i.e., occupy the same node)

Cop Number of a graph G

$cn(G)$: min # Cops to win in G



Link with graph structure? a first surprising (?) example

$cn(G) \leq 3$ for any planar graph G (based on decomposition with shortest paths)

[Aigner and Fromme 84]

Cops and Robber vs. Graph Structure

Link with Structural Properties of n-node Graphs

Large girth (smallest cycle) AND large min degree $\Rightarrow$ large cop-number [Frankl 87]

$\Rightarrow \exists$ n-node graphs G with $cn(G) = \Omega(\sqrt{n})$ (e.g., random $\sqrt{n}$-regular graphs)

cn in general n-node graphs

Conjecture: $cn(G) = \Theta(\sqrt{n})$ [Meyniel 85]

Upper bound: $\frac{n}{2^{(1-o(1))\sqrt{\log n}}} \geq n^{1-\epsilon}$ for any ϵ [Scott, Sudakov 11, Lu,Peng 12]

Meyniel Conjecture TRUE in many graph classes

	cn	
dominating set $\leq k$	$\leq k$	[folklore]
treewidth $\leq t$	$\leq t/2+1$	[Joret, Kaminski,Theis 09]
genus $\leq g$	$\leq \lfloor \frac{3g}{2} \rfloor + 3$	(conjecture $\leq g+3$) [Schröder, 01]
H**-minor free**	$\leq \lvert E(H) \rvert$	[Andreae, 86]
degeneracy $\leq d$	$\leq d$	[Lu,Peng 12]
diameter 2	$O(\sqrt{n})$	–
bipartite diameter 3	$O(\sqrt{n})$	–
random graphs	$O(\sqrt{n})$	[Bollobas *et al.* 08] [Luczak, Pralat 10]

Link with Structural Properties of n-node Graphs

Large girth (smallest cycle) AND large min degree $\Rightarrow$ large cop-number [Frankl 87]

$\Rightarrow \exists$ n-node graphs G with $cn(G) = \Omega(\sqrt{n})$ (e.g., random $\sqrt{n}$-regular graphs)

cn in general n-node graphs

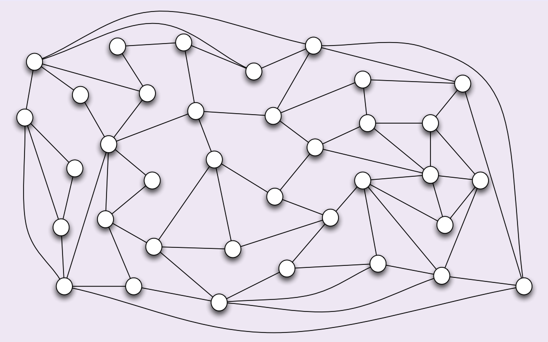
Conjecture: $cn(G) = \Theta(\sqrt{n})$ [Meyniel 85]

Upper bound: $\frac{n}{2^{(1-o(1))\sqrt{\log n}}} \geq n^{1-\epsilon}$ for any ϵ [Scott, Sudakov 11, Lu,Peng 12]

Meyniel Conjecture TRUE in many graph classes

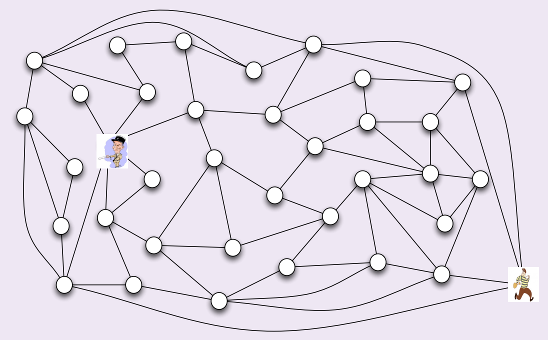
	cn	
dominating set $\leq k$	$\leq k$	[folklore]
treewidth $\leq t$	$\leq t/2+1$	[Joret, Kaminski,Theis 09]
genus $\leq g$	$\leq \lfloor \frac{3g}{2} \rfloor + 3$	*(conjecture $\leq g+3$)* [Schröder, 01]
H**-minor free**	$\leq \lvert E(H) \rvert$	[Andreae, 86]
degeneracy $\leq d$	$\leq d$	[Lu,Peng 12]
diameter 2	$O(\sqrt{n})$	–
bipartite diameter 3	$O(\sqrt{n})$	–
random graphs	$O(\sqrt{n})$	[Bollobas *et al.* 08] [Luczak, Pralat 10]

From Meyniel Conjecture in k-chordal Graphs...



A simple universal strategy (Cops must occupy an **induced** path)

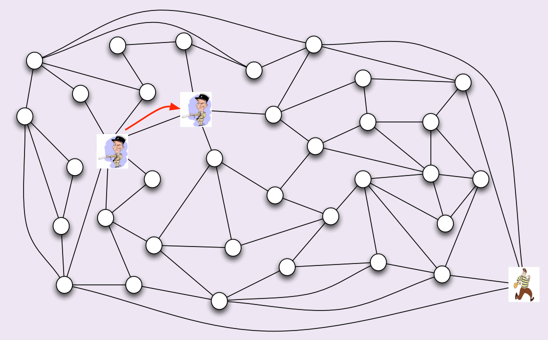
From Meyniel Conjecture in k-chordal Graphs...



A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node

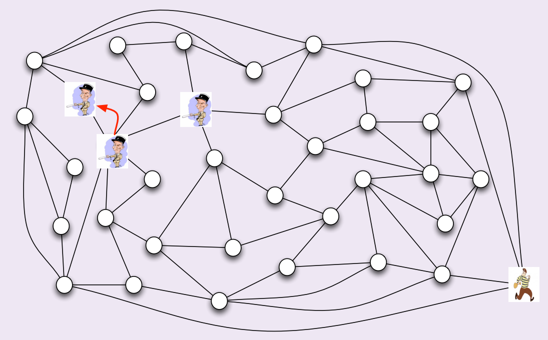
From Meyniel Conjecture in k-chordal Graphs...



A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path**

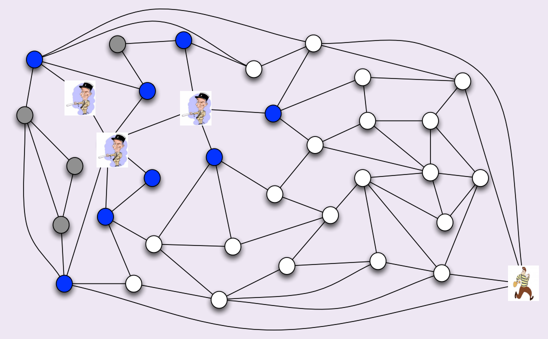
From Meyniel Conjecture in k-chordal Graphs...



A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path**

From Meyniel Conjecture in k-chordal Graphs...

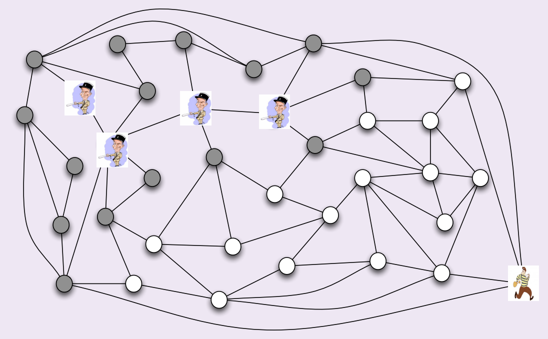


A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path**

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")

From Meyniel Conjecture in k-chordal Graphs...

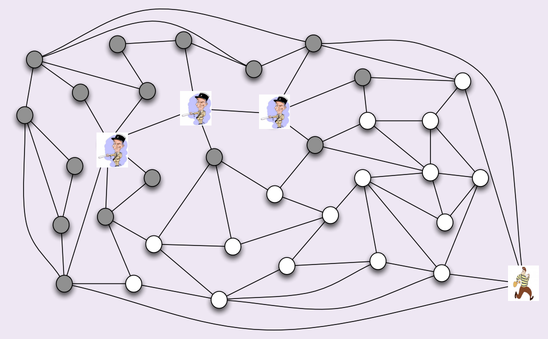


A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path**

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")

From Meyniel Conjecture in k-chordal Graphs...

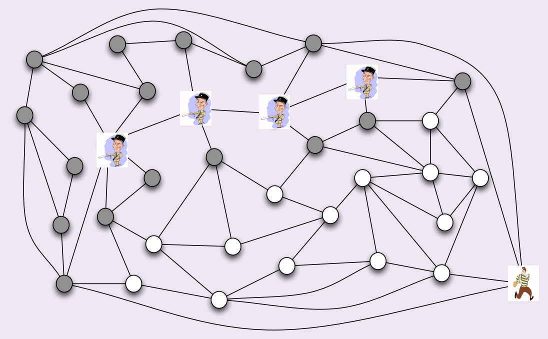


A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path** (3) "retract" when useless

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")

From Meyniel Conjecture in k-chordal Graphs...

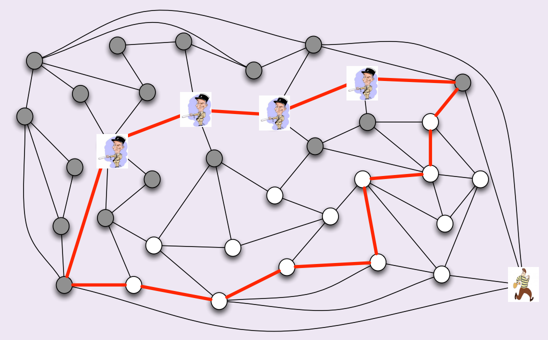


A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path** (3) "retract" when useless

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")

From Meyniel Conjecture in k-chordal Graphs...

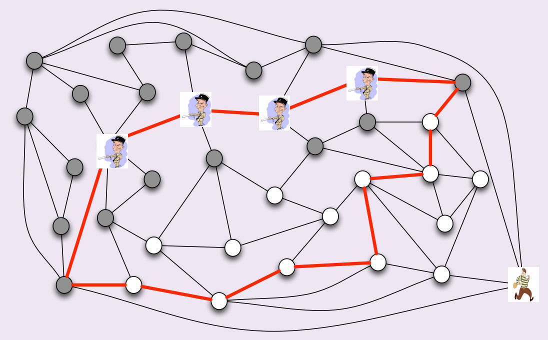


A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path** (3) "retract" when useless

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")
Key point 2: use k Cops only if there is an induced cycle of length $\geq k+1$

From Meyniel Conjecture in k-chordal Graphs...



A simple universal strategy (Cops must occupy an **induced** path)

(1) start in a node (2) greedily extend along **induced path** (3) "retract" when useless

Key point 1: aim at Neighborhood[Cops] induces a separator (grey nodes "protected")
Key point 2: use k Cops only if there is an induced cycle of length $\geq k+1$

Theorem [Kosowski,Li,N.,Suchan, ICALP'12, Algorithmica14]

$cn(G) \leq k-1$ in any graph G with maximum induced cycle of length k (k-chordal)

...to a Structural Result and Applications to Compact Routing

Recursive decomposition using **separators** with short **dominating induced path**

Theorem

There is a $O(m^2)$ algorithm that, for any graph G with m edges and max degree Δ,

- either returns an induced cycle of length $\geq k+1$,
- or compute a tree-decomposition with width $\leq (k-1)(\Delta-1)+2$.

Corollary: $tw(G) = O(\Delta \cdot k)$ if G has no induced cycle of length $> k$.

Compact routing scheme in k-chordal graphs

additive stretch: $O(k \log \Delta)$, Routing Tables: $O(k \log n)$ bits.
use bags as "shortcut"

[Kosowski,Li,N.,Suchan, ICALP'12, Algorithmica14]

Complex networks $\Rightarrow$ high clustering coefficient $\Rightarrow$ "few" large induced cycles

Variant of Cops and Robber vs. Graph Structure

When Cops and Robber can run

New variant with speed: Players may move along several edges per turn
$cn_{s',s}(G)$: min # of Cops with speed s' to capture Robber with speed s, $s \geq s'$.

Meyniel Conjecture [Alon, Mehrabian'11] and general upper bound [Frieze,Krivelevich,Loh'12] extend to this variant

... but fundamental differences (recall: planar graphs have $cn_{1,1} \leq 3$)

$cn_{1,2}(G)$ unbounded in grids [Fomin,Golovach,Kratochvil,N.,Suchan TCS'10]

Open question: $\Omega(\sqrt{\log n}) \leq cn_{1,2}(G) \leq O(n)$ in $n \times n$ grid G exact value?

When Cops and Robber can run

G is Cop-win $\Leftrightarrow$ 1 Cop sufficient to capture Robber in G

Structural characterization of Cop-win graphs for any speed s and s'

[Chalopin,Chepoi,N.,Vaxès SIDMA'11]

generalize seminal work of [Nowakowski,Winkler'83]

hyperbolicity δ of G: measures the "proximity" of the metric of G with a tree metric

New characterization and algorithm for hyperbolicity

- bounded hyperbolicity $\Rightarrow$ one Cop can catch Robber almost twice faster

  [Chalopin,Chepoi,N.,Vaxès SIDMA'11]
- one Cop can capture a faster Robber $\Rightarrow$ bounded hyperbolicity

  [Chalopin,Chepoi,Papasoglu,Pecatte 14]

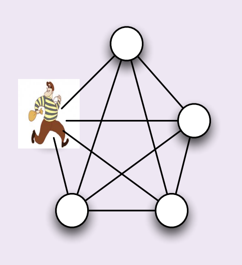
$\Rightarrow$ $O(1)$-approximation sub-cubic-time for hyperbolicity [Chalopin,Chepoi,Papasoglu,Pecatte 14]

Outline

Visible Graph Searching

[Seymour,Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).



Visible search Number $vs(G)$: # min of Cops.

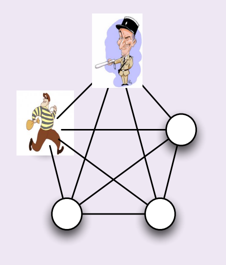
Very different from Cops & Robber

e.g., $vs(K_n) = n$ (while $cn(K_n) = 1$)

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).



Visible search Number $vs(G)$: # min of Cops.

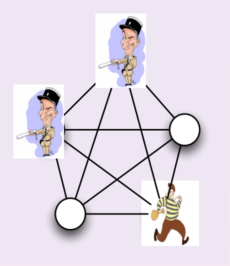
Very different from Cops & Robber

e.g., $vs(K_n) = n$ (while $cn(K_n) = 1$)

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).



Visible search Number $vs(G)$: # min of Cops.

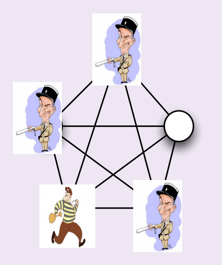
Very different from Cops & Robber

e.g., $vs(K_n) = n$ (while $cn(K_n) = 1$)

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).



Visible search Number $vs(G)$: # min of Cops.

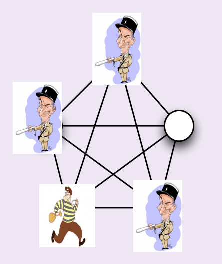
Very different from Cops & Robber

e.g., $vs(K_n) = n$ (while $cn(K_n) = 1$)

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).



Visible search Number $vs(G)$: # min of Cops.

Very different from Cops & Robber

e.g., $vs(K_n) = n$ (while $cn(K_n) = 1$)

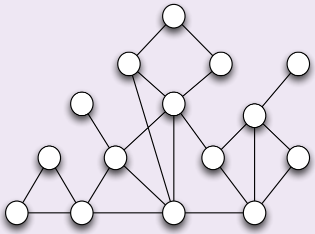
Graph Searching as algorithmic interpretation of Decompositions

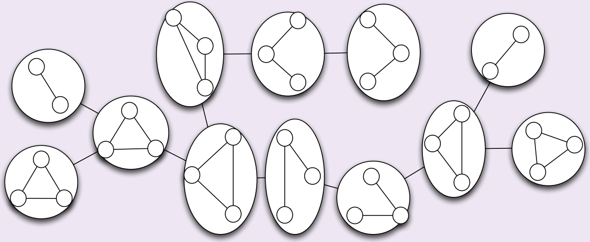
For any graph G, $vs(G) = tw(G) + 1$ [Seymour, Thomas 93]
tree-decomposition of width $k \Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber

Visible Graph Searching

[Seymour,Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





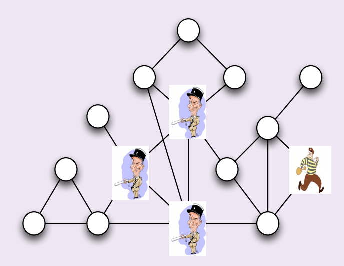
Graph Searching as algorithmic interpretation of Decompositions

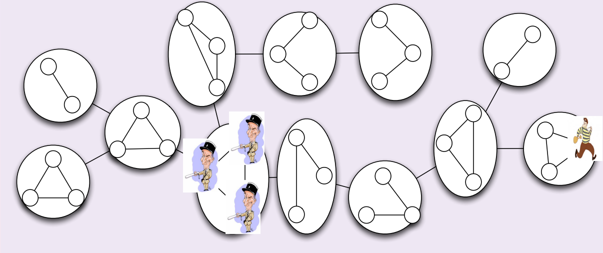
For any graph G, $vs(G) = tw(G) + 1$ [Seymour,Thomas 93]
tree-decomposition of width $k \Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





Graph Searching as algorithmic interpretation of Decompositions

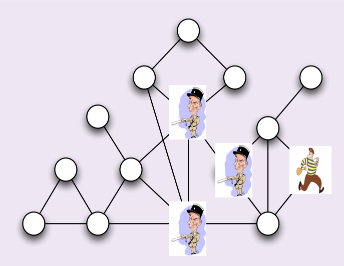
For any graph G, $vs(G) = tw(G) + 1$ [Seymour, Thomas 93]

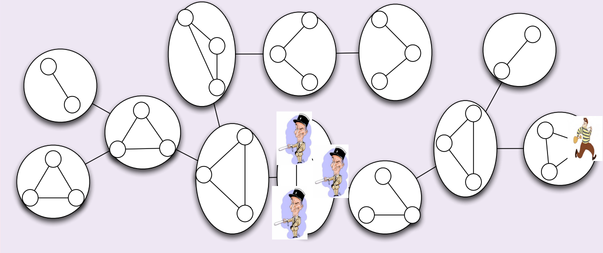
tree-decomposition of width $k \Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





Graph Searching as algorithmic interpretation of Decompositions

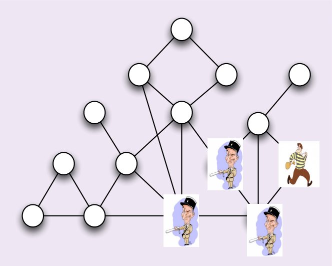
For any graph G, $vs(G) = tw(G) + 1$ [Seymour, Thomas 93]

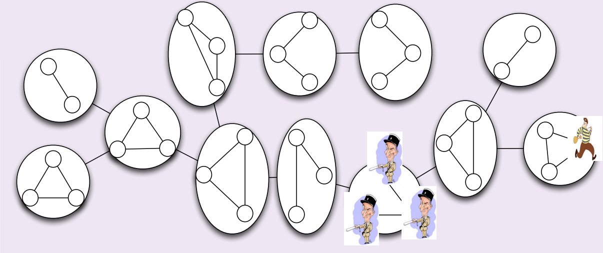
tree-decomposition of width k $\Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber

Visible Graph Searching

[Seymour,Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





Graph Searching as algorithmic interpretation of Decompositions

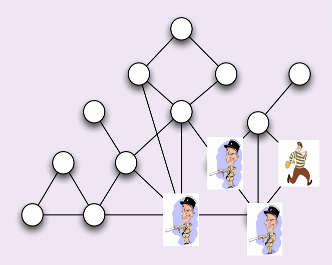
For any graph G, $vs(G) = tw(G) + 1$ [Seymour,Thomas 93]

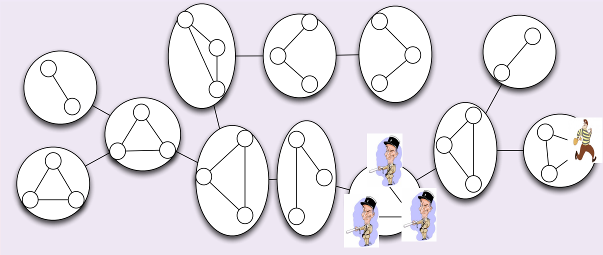
tree-decomposition of width k $\Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber

Visible Graph Searching

[Seymour, Thomas 93]

- **Visible** Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





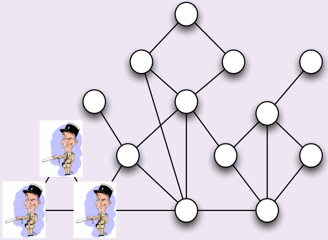
Graph Searching as algorithmic interpretation of Decompositions

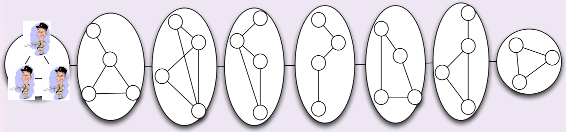
For any graph G, $vs(G) = tw(G) + 1$ [Seymour, Thomas 93]
tree-decomposition of width $k \Leftrightarrow$ strategy with $k+1$ cops vs. **visible** Robber
based on duality result: $tw(G) + 1 =$ max order of *bramble* in G

InVisible Graph Searching

[Breish 67, Parsons 78]

- InVisible Robber moves arbitrarily fast, at any time, while not crossing cops;
- Cops can be **Placed** or **Removed** till Robber is captured (and cannot flee).





Graph Searching as algorithmic interpretation of Decompositions

For any graph G, $s(G) = pw(G) + 1$ [Bienstock,Seymour 91]
path-decomposition of width $k \Leftrightarrow$ strategy with $k+1$ cops vs. **invisible** Robber.

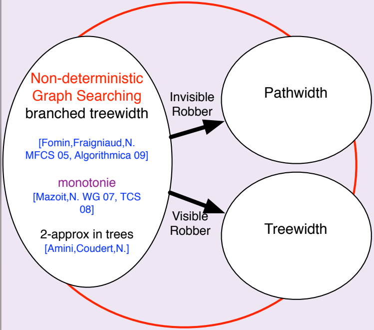
Understand Width Parameters
thanks to
Graph Searching Games

Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times

[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\qquad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth

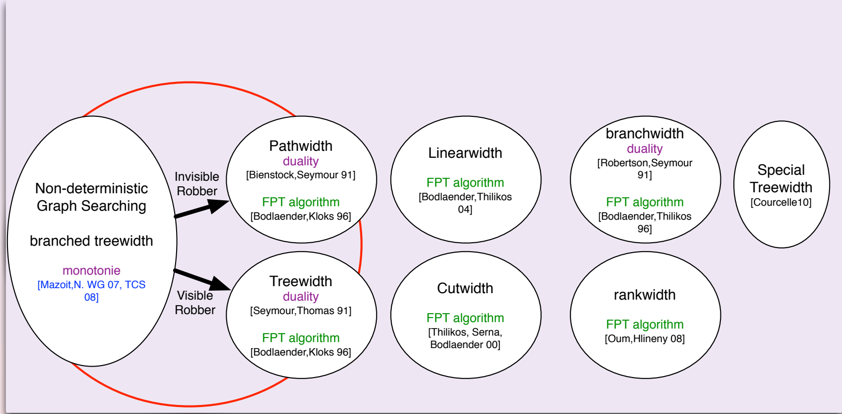


Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times

[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

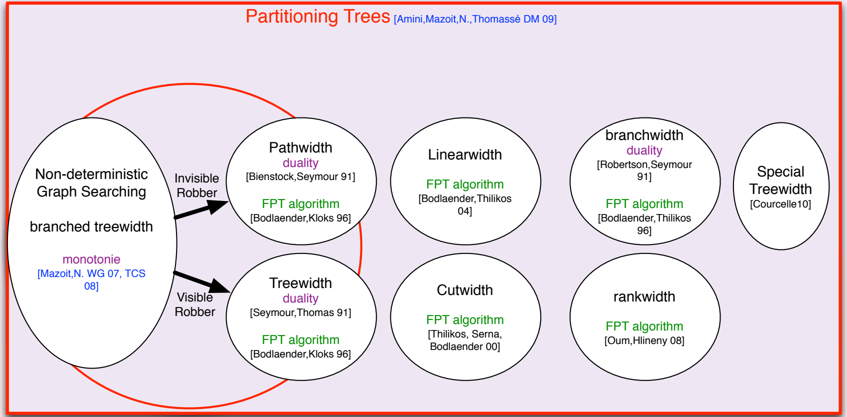
$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\quad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth



Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times
[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\qquad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth

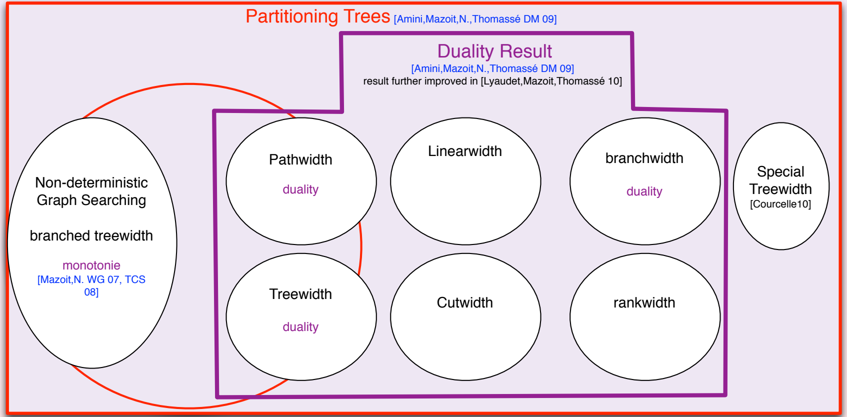


Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times

[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\qquad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth

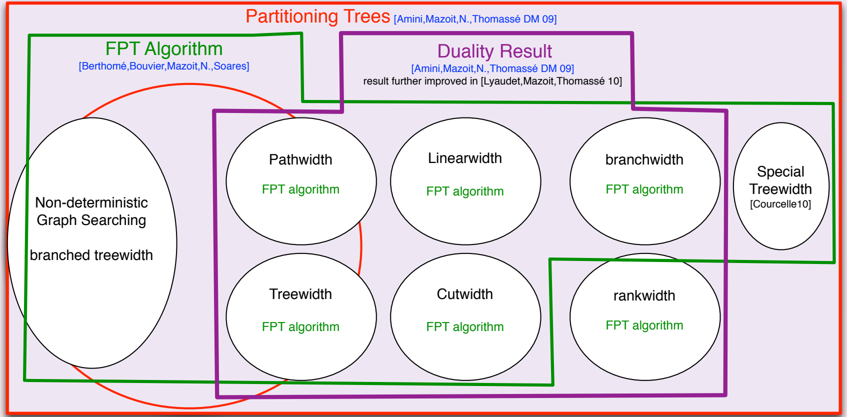


Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times

[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

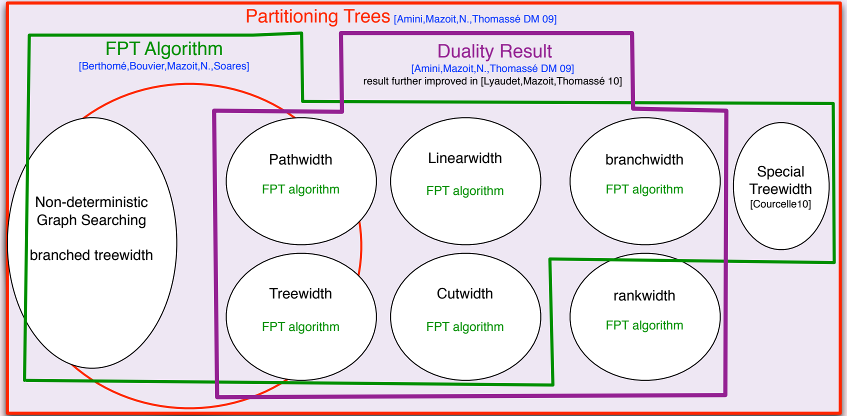
$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\quad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth



Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times
[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\quad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth



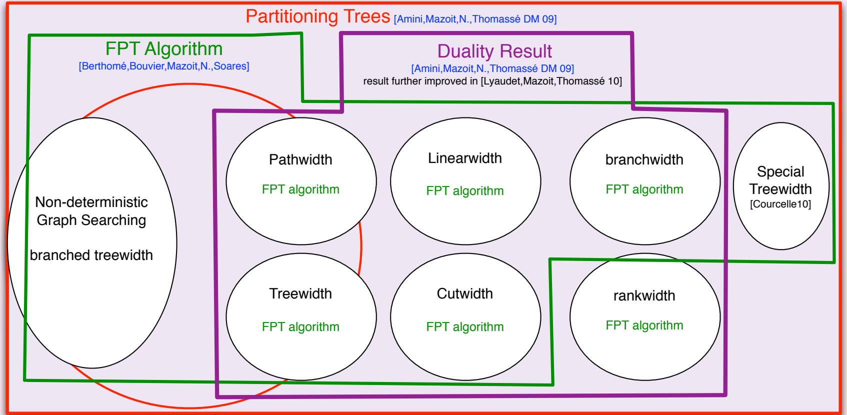
open problem 1: what about directed graphs?

Non-deterministic GS: unified graph decomposition

Non-deterministic Graph Searching: Cops can see Robber at most $q \in \mathbb{N}$ times

[Fomin,Fraigniaud,N., MFCS 05, Algorithmica 09]

$q = 0 \Leftrightarrow$ Invisible Robber $\Leftrightarrow$ Pathwidth $\quad q = \infty \Leftrightarrow$ Visible Robber $\Leftrightarrow$ Treewidth



open problem 2: what about actual computation of decompositions?

Outline

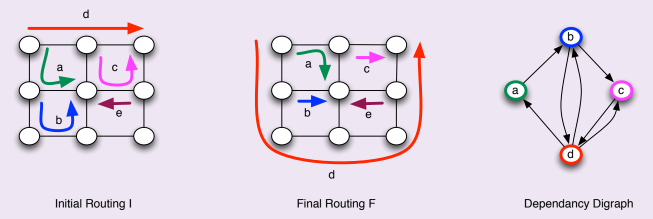
1. Cops and Robber Games
   - Rules of the game
   - k-chordal Graphs and Routing
   - Fast Cops vs. fast Robber
2. Graph Searching
   - Graph Searching and Graph Decompositions
   - New Approach for Width Parameters
3. Games to model Telecommunication Problems
   - Graph Searching and Routing Reconfiguration
   - Turn-by-turn Game for Prefetching
4. Conclusion and Perspective
   - Conclusion and other Contributions
   - Perspectives

Routing Reconfiguration in WDM Networks

Graph Searching as a model for scheduling problems

Switching routes of requests, one by one, disturbing the traffic as few as possible

[Coudert,Pérennes,Pham,Sereni'05]



complexity, tradeoffs, algorithms, physical constraints...

[Solano,Pioro'13] [Coudert,Huc,Mazauric,N.,Sereni ONDM'09] [Cohen,Coudert,Mazauric,Nepomuceno,N. FUN'10,TCS'11]...

New path-decomposition for directed graphs [N.,Soares LAGOS'13]
further work: corresponding digraph tree-decomposition?

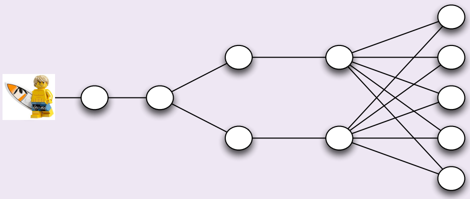
A Turn-by-turn Game to model Prefetching

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task

Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



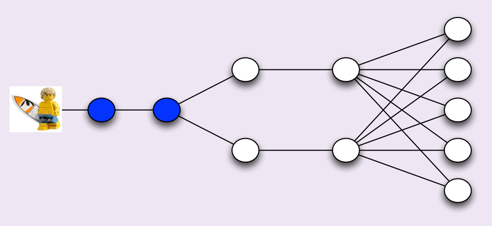
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



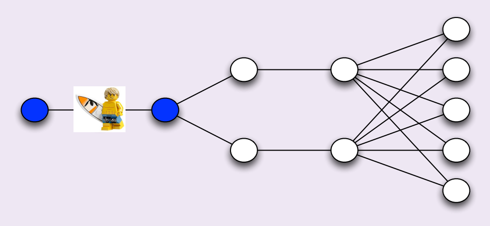
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



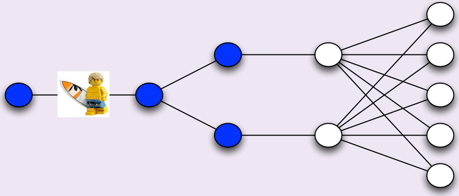
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



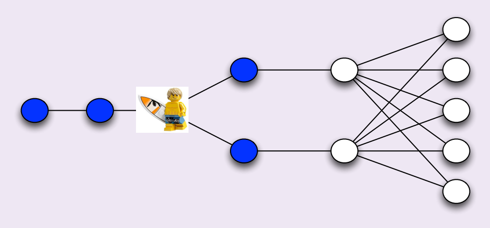
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



Initially, Web-surfer at some (given) node, and **Turn-by-turn**

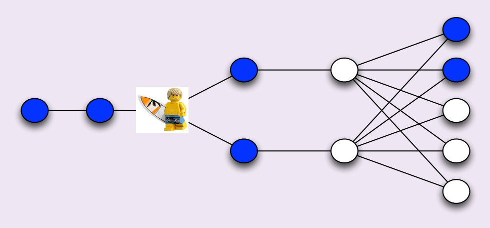
1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task

Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



Initially, Web-surfer at some (given) node, and **Turn-by-turn**

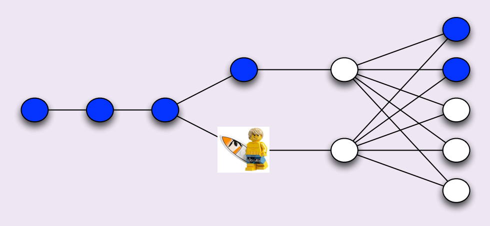
1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task

Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



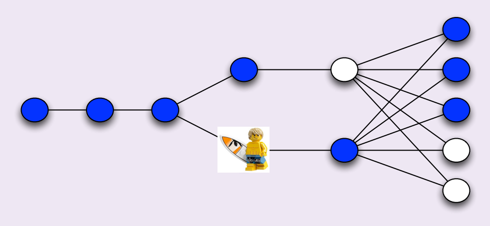
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



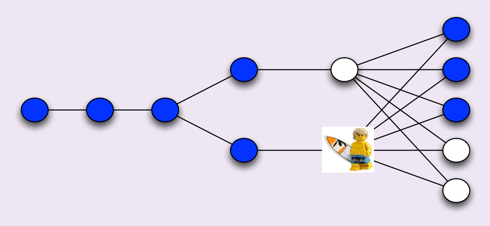
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]



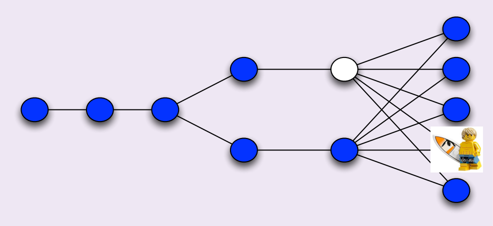
Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

Prefetching and Surveillance Game

Model for Prefetching/Caching

Parallelism between execution of one task and **transfer** of information necessary to next task Surveillance game: [Fomin,Giroire,Jean-Marie,Mazauric,N.]

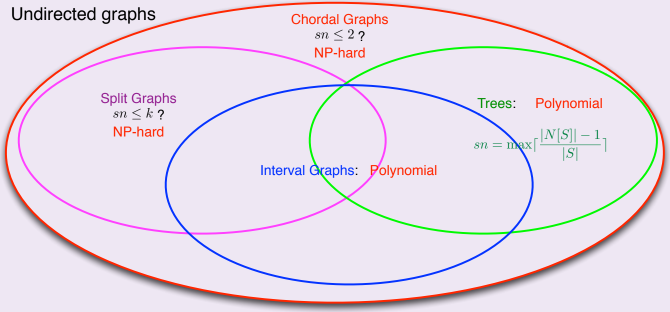


Initially, Web-surfer at some (given) node, and **Turn-by-turn**

1. Web-browser prefetches $\leq k$ pages, i.e., marks $\leq k$ nodes
2. Web-surfer may move on adjacent node

surveillance number(G, v_0) = min. number k of marks per turn avoiding Surfer (starting from v_0) to reach an unmarked node (in the example = 2)

Surveillance game: results and open problems



Directed graphs	General graphs
DAGs: $sn \leq 4$? PSPACE-complete	$degree(v_0) \leq sn \leq \max\{degree(v_0); \Delta - 1\}$
DAGs max degree 6: $sn \leq 2$? NP-hard	$O(4^n)$ exact algorithm

[Fomin,Giroire,Jean-Marie,Mazauric,N. FUN'12,TCS'13]

Online version: best strategy: $\Theta(\Delta)$ marks per turn [Giroire,N.,Pérennes,Soares SIROCCO'13]

Outline

Conclusion

Pursuit-Evasion games ⇒ interesting point of view

- for understanding/exploiting/discovering graph structural properties
- for modeling and studying optimization problems...

Other contributions related to optimization and graphs' structure (No Cops!)

- Weighted Coloring is not in P in trees unless ETH fails [Araújo,N.,Pérennes STACS'14]
- Convexity in some graph classes. [Araújo,Campos,Giroire,N.,Sampaio,Soares TCS'13]
- Gathering in wireless grid networks with interference [Bermond,Li,N.,Rivano,Yu]

Perspective: Computation of Graph Decompositions

Difficult to compute but some good news

- constant approximation for treewidth in planar graphs [Seymour,Thomas'94]
- $O(\sqrt{\log OPT})$-approximation for treewidth (using SDP) [Feige,Hajiaghayi,Lee'05]

However, a lot remains unknown...

- complexity of treewidth in planar graphs?
- constant approximation for pathwidth/treewidth?

Moreover, almost nothing in practice...

- heuristics [Bodlaender,Koster'10]
- Branch & Bound for treewidth [QuickBB]
- Branch & Bound for pathwidth (up to $\approx$ 70 nodes) [Coudert,Mazauric,N., SEA'14]

$\Rightarrow$ Lack of Lower bounds

Approximations? Using new graph searching games:

Connected Graph Searching [Dereniowski]
Exclusive Graph Searching [Blin,Burman,N. ESA'13]

Perspective: Fractional Games

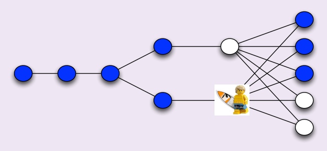
Turn-by-turn games may be even harder:

- Maker and Breaker: EXPTIME-complete (when decidable) [Arul,Reichert 13]
- Cops and Robber: EXPTIME-complete [Kinnersley 14]
- Surveillance game: PSPACE-complete [Fomin,Giroire,Jean-Marie,Mazauric,N., TCS'13]
- Eternal Vertex Cover: NP-hard [Fomin,Gaspers,Golovach,Kratsch,Saurabh 10]

Flashback to Surveillance game

"close" to sequential instances of Hitting Set

What about a fractional relaxation?
$\Rightarrow$ fractions of nodes can be marked at each step



On going work: exponential algorithm (LP) for Fractional Surveillance game [Giroire,N.,Pérennes,Soares]

Hopes: logarithmic fractional gap (random rounding?),

apply same relaxation to approximate Graph Searching games/decompositions?

Perspective: Large Scale Networks

Large Scale Networks: only partially known

Title of the HDR: "Algorithmic Complexity: Between Structure and Knowledge"
What about "Knowledge"?

How to use small local knowledge to compute global properties: structure also helps

[Becker, Kosowski,Matamala,N.,Rapaport,Suchan,Todinca IPDPS'11,SPAA'12,Distributed Computing'14]

How structure helps to design distributed/localized algorithms

- Distributed Graph Searching and Models for Mobile Agent Computing [Ilcinkas,N.,Soguet Distributed Computing'09] [d'Angelo,DiStefano,Navarra,N.,Suchan Algorithmica'14]...
- Fault-tolerant routing in paths and expanders [Hanusse,Ilcinkas,Kosowski,N., PODC'10]
- Diffusion in P2P networks [Giroire,Modrzejewski,N.Pérennes SIROCCO'13]

Other perspectives: Distributed/local computation in large scale networks

Merci de votre attention !

Graph Searching to approximate Pathwidth?

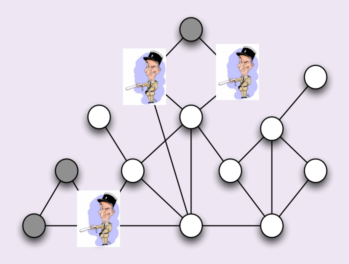
Connected Graph Searching

"cleared" area must be always connected
Connected search number $cs(G)$: # min of Cops

$\forall$ graph G, $cs(G) \leq 2s(G) + O(1)$ [Dereniowski SIDMA'12]

non monotone [Yang,Dyer,Alspach DM'09]

- open question: in NP?
- open question: FPT?



example of non-connected step

3-approximation for cs in weighted trees [Dereniowski TCS'12]

pw is NP-hard in weighted trees [Mihai,Todinca FAW'09]

on going work: chordal graphs?

Graph Searching to approximate pathwidth?

Exclusive Graph Searching

new constraint: at most one Cop per node at every step [Blin,Burman,N., ESA'13]
(Cops can slide along edges)

$xs(G)$: min # of Cops $mxs(G)$: min # of Cops for monotone strategies

variant not monotone ($xs(G)$ may differ from $mxs(G)$) [Blin,Burman,N., ESA'13]
For any graph G with max. degree Δ, $s(G) \leq xs(G) \leq (\Delta - 1)(s(G) + 1)$

About complexity: Computing xs is

- NP-hard in planar graphs with max degree 3 [Markou,N.,Pérennes]
- polynomial in trees [Blin,Burman,N. ESA'13]
- linear in cographs [Markou,N.,Pérennes]

Graph Searching to approximate pathwidth?

Exclusive Graph Searching

new constraint: at most one Cop per node at every step [Blin,Burman,N., ESA'13]
(Cops can slide along edges)

$xs(G)$: min # of Cops $mxs(G)$: min # of Cops for monotone strategies

variant not monotone ($xs(G)$ may differ from $mxs(G)$) [Blin,Burman,N., ESA'13]
For any graph G with max. degree Δ, $s(G) \leq xs(G) \leq (\Delta - 1)(s(G) + 1)$

About complexity: Computing xs is

- NP-hard in planar graphs with max degree 3 [Markou,N.,Pérennes]
- polynomial in trees [Blin,Burman,N. ESA'13]
- linear in cographs [Markou,N.,Pérennes]

	pathwidth [Gustedt'93]	monotone exclusive-search [Markou,N.,Pérennes]
split graphs	**P**	**NP-complete**
star-like graphs with ≥ 2 peripheral nodes per clique	**NP-complete**	**P**

Graph Searching to approximate pathwidth?

Further Work:

- Are there graph classes where *pw* is NP-complete
  and *xs* (*mxs*) in *P* and provide good approximation of *pw*?
  (or *vice-versa*)
- Can *xs* (or *mxs*) be approximated?
- *xs* in NP?
- *xs* (or *mxs*) FPT?