k-Chordal Graphs: from Cops and Robber to Compact Routing via Treewidth[1]

Adrian Kosowski[1] Bi Li[2,3] Nicolas Nisse[2]
Karol Suchan[4,5]

[1] CEPAGE, INRIA, Univ. Bordeaux 1, France
[2] MASCOTTE, INRIA, I3S (CNRS, UNS) Sophia Antipolis, France
[3] AMSS, CAS, China
[4] Univ. Adolfo Ibáñez, Facultad de Ingenieria y Ciencias, Santiago, Chile
[5] WMS, AGH - Univ. of Science and Technology, Krakow, Poland

Seminario, Univ. Adolfo Ibáñez, Santiago, August 2012

[1] presented to ICALP 2012

Outline

1. Distributed and Compact Routing
   - Routing in a distributed way
   - Compact Routing
   - Using structural Properties: chordality
   - Our results

2. Cops and Robber Games
   - Definitions and related works
   - Cops and Robber in k-chordal graphs

3. Structural result and link with compact routing

What is Routing?

Something/someone is at some node (**the source**) of a network

(city, country, Internet, etc.)

e.g., you at home or in a city,

an email that you want to send in your computer,

etc.

and needs to go to another node (**the destination**)

you want to go somewhere,

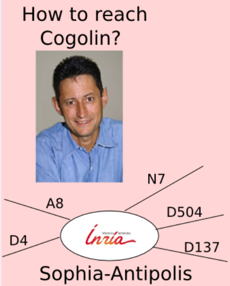
to send the email to someone,

etc.

Goal: to reach the destination quickly.

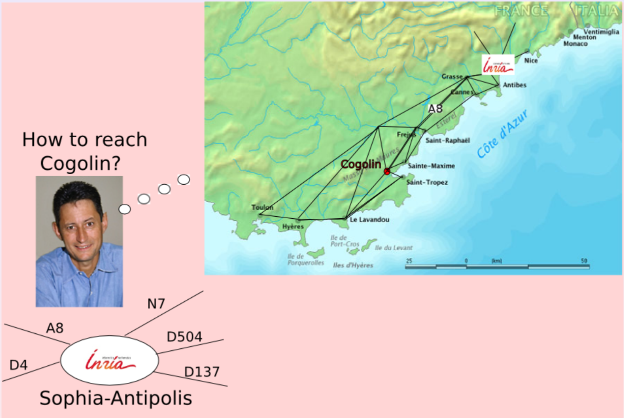
Why may Routing be difficult?

Jean-Claude wants to reach his destination



Why may Routing be difficult?

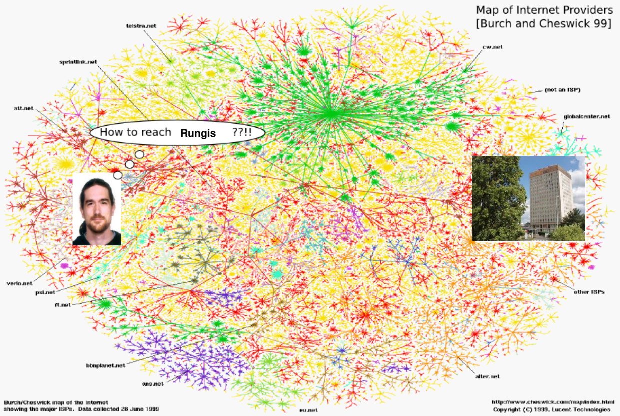
Jean-Claude wants to reach his destination



If the network is **small**, **known**, **static**, etc. Easy !!
Apply your favorite shortest path algorithm (e.g., Dijkstra)

Why may Routing be difficult?

What for David?

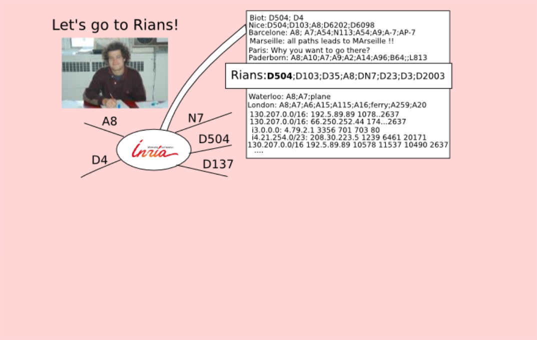


If the network is **Huge**, **only partially known**, **dynamic**, etc.

What to do ??

How Internet works? Border Gateway Protocol

BGP: routing protocol of the Autonomous Systems' (AS) network



- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored (to avoid loops)

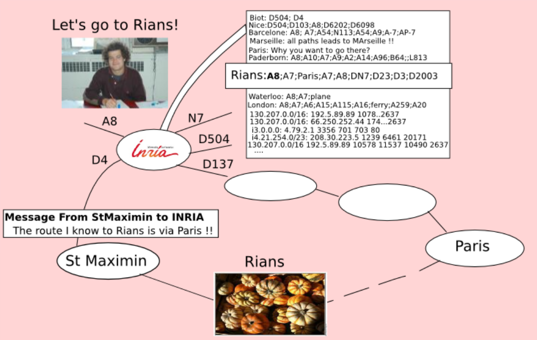
How Internet works? Border Gateway Protocol

BGP: routing protocol of the Autonomous Systems' (AS) network

- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored $\Rightarrow$ huge, difficult to read

How Internet works? Border Gateway Protocol

BGP: routing protocol of the Autonomous Systems' (AS) network



- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored $\Rightarrow$ huge, difficult to read
- to deal with dynamicity: ASs send to each other paths they know

ASs may lie (Policy), paths may be too long

Challenges

	BGP	Ideally
Routing Tables size	$O(n \log n)$ bits	$O(\log n)$ bits
Paths Length	depend on ASs policies	Shortest paths
Update Time	Long ($\approx$ 5 min.)	Fast (using only local information)

(n is the number of ASs)

Large-scale Networks have specific structural properties

- small diameter, high clustering coefficient, power-law degree-distribution
- hyperbolicity, chordality, etc.

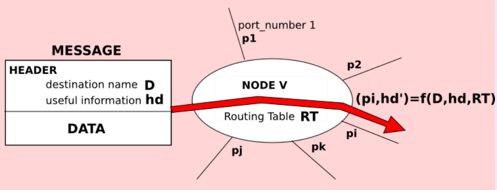
Objectives

- Understand (find new) Properties
- Use it for algorithmic purposes (not only routing)
- Model such networks
- Simulate (static/dynamic behavior)

Compact Routing

Goal To deliver a message in a distributed way

Routing Scheme protocol that directs the traffic in a network



Routing Problem: definitions of:

- Routing Tables **RT**
- Information required in the message headers **hd**
- Routing function **f**: compute next hop / may modify the header

Models

- **labelled/name independent** node Identifiers are part of the design or not
- **design port model** local labeling (port number) are part of the design or not

Performances Measures

Stretch

How far the route actually followed is from a shortest path

- **Multiplicative stretch**: $\max_{x,y \in V(G)} \leq \frac{|route(x,y)|}{dist(x,y)}$.
- **Additive stretch**: $\max_{x,y \in V(G)} \leq |route(x,y)| - dist(x,y)$.

Memory space

- Space necessary to store local routing table (per node)
- Size of the node Identifiers / message header (generally $O(\log n)$)

Time complexity

- Distributed/Centralized protocol
- Time to setup data structures
- Time to update data structures

Example: Interval Routing [Santoro & Khatib, 82]

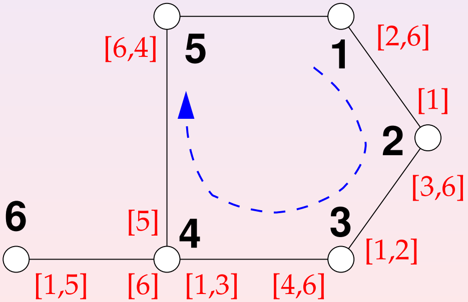
- Nodes labeled using integers
- Outgoing arc labeled with an interval of the name range

Message sent through the arc containing the destination

mult-stretch:
$\frac{route(1,5)}{d(1,5)} = 4$

add-stretch:
$route(1,5)\text{-}d(1,5) = 3$

space per node:
$O(\Delta \log n)$



Related Works

Names and Headers (if any) are of polylogarithmic size

network	mult. stretch	routing table	
		labelled	name-independent
arbitrary ($k \geq 2$)	shortest path	$O(n \log n)$ [folk]	$\Theta(n \log n)$ [Gavoille,Pérennes]
	$O(k)$	$O(n^{1/k})$ [Thorup,Zwick]	$\Theta(n^{1/k})$ [TZ/Abraham et al.]
trees	shortest path	$O(\log n)$ [TZ/Fraigniaud,Gavoille]	$\Omega(\sqrt{n})$ [Laing,Rajaraman]
	$2^k - 1$		$\Theta(n^{1/k})$ [Laing/Abraham et al.]
doubling-α dimension	$O(1) + \epsilon$	$O(\log \Delta)$ [Talwar/Slivkins]	$O(\epsilon^{-\alpha} \log n)$ [Abraham et al.]
		$O(\log n)$ [Chan et al./Abraham et al.]	
planar	$1 + \epsilon$	$O(\log n)$ [Thorup]	
H-minor free	$1 + \epsilon$	$O(\|H\|! \cdot 2^{\|H\|} \log n)$ [Abraham,Gavoille]	

In general graphs, $\Theta(n \log n)$ is optimal.

Can we do better than BGP?

Yes !! (we hope), using structural properties

Related Works

Names and Headers (if any) are of polylogarithmic size

network	mult. stretch	routing table	
		labelled	name-independent
arbitrary	shortest path	$O(n \log n)$ [folk]	$\Theta(n \log n)$ [Gavoille,Pérennes]
$(k \geq 2)$	$O(k)$	$O(n^{1/k})$ [Thorup,Zwick]	$\Theta(n^{1/k})$ [TZ/Abraham et al.]
trees	shortest path	$O(\log n)$ [TZ/Fraigniaud,Gavoille]	$\Omega(\sqrt{n})$ [Laing,Rajaraman]
	$2^k - 1$		$\Theta(n^{1/k})$ [Laing/Abraham et al.]
doubling-α dimension	$O(1) + \epsilon$	$O(\log \Delta)$ [Talwar/Slivkins]	$O(\epsilon^{-\alpha} \log n)$ [Abraham et al.]
		$O(\log n)$ [Chan et al./Abraham et al.]	
planar	$1 + \epsilon$	$O(\log n)$ [Thorup]	
H-minor free	$1 + \epsilon$	$O(\lvert H\rvert! \cdot 2^{\lvert H\rvert} \log n)$ [Abraham,Gavoille]	

In general graphs, $\Theta(n \log n)$ is optimal.

Can we do better than BGP?

Yes !! (we hope), using structural properties

Properties of large scale networks — Chordality

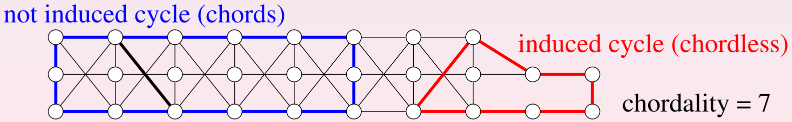
Well known properties	graph parameters
small diameter (logarithmic)	($\Rightarrow$ small hyperbolicity)
power law degree distribution	
high clustering coefficient	$\Rightarrow$ **few long induced cycles**

Properties of large scale networks — Chordality

Well known properties — graph parameters

small diameter (logarithmic) ($\Rightarrow$ small hyperbolicity)
power law degree distribution
high clustering coefficient $\Rightarrow$ **few long induced cycles**

Chordality of a graph G: length of **greatest induced cycle** in G



Brief related work on chordality

Complexity — chordality $\leq k$?

NP-complete	easy reduction from hamiltonian cycle
not FPT [CF'07]	no algorithm $f(k).poly(n)$ (unless $P = NP$)
FPT in planar graphs [KK'09]	Graph Minor Theory

chordality $\leq k \Rightarrow$ treewidth $\leq O(\Delta^k)$ [Bodlaender, Thilikos'97]

Compact routing schemes in graphs with chordality $\leq k$

stretch	RT's size	computation time	
$k+1$	$O(k \log^2 n)$	$poly(n)$	[Dourisboure'05]
header never changes			
$k-1$	$O(\Delta \log n)$	$O(D)$	[NRS'09]
distributed protocol to compute RT's / no header			
$O(k \log \Delta)$	$O(k \log n)$	$O(m^2)$	**[this paper]**
or $O(k)$ and $O(\Delta \log n)$		generalization of [NRS'09]	

Names and Headers (if any) are of polylogarithmic size

Brief related work on chordality

Complexity chordality $\leq k$?

NP-complete — easy reduction from hamiltonian cycle
not FPT [CF'07] — no algorithm $f(k).poly(n)$ (unless $P = NP$)
FPT in planar graphs [KK'09] — Graph Minor Theory

chordality $\leq k \Rightarrow$ treewidth $\leq O(\Delta^k)$ [Bodlaender, Thilikos'97]

Compact routing schemes in graphs with chordality $\leq k$

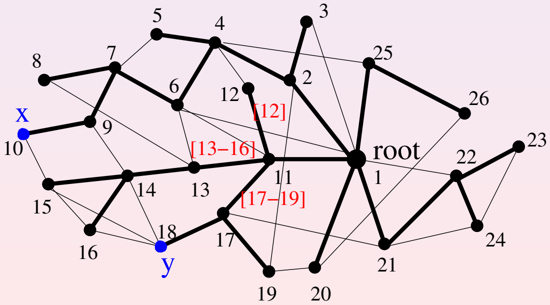
stretch	RT's size	computation time	
$k+1$	$O(k \log^2 n)$	$poly(n)$	[Dourisboure'05]
header never changes			
$k-1$	$O(\Delta \log n)$	$O(D)$	[NRS'09]
distributed protocol to compute RT's / no header			
$O(k \log \Delta)$	$O(k \log n)$	$O(m^2)$	**[this paper]**
or $O(k)$ and $O(\Delta \log n)$		generalization of [NRS'09]	

Names and Headers (if any) are of polylogarithmic size

Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

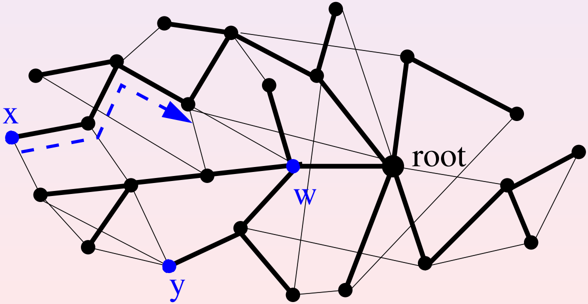


Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

> If $x = y$, stop.
> If there is $w \in N_G(x)$, an ancestor of y in T,
> choose w minimizing $d_T(w, y)$;
> Otherwise, choose the parent of x in T.

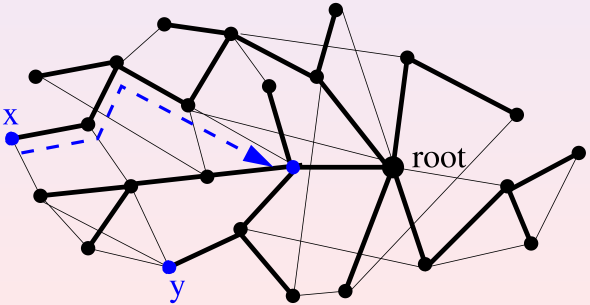


Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

> If $x = y$, stop.
> If there is $w \in N_G(x)$, an ancestor of y in T,
> choose w minimizing $d_T(w, y)$;
> Otherwise, choose the parent of x in T.

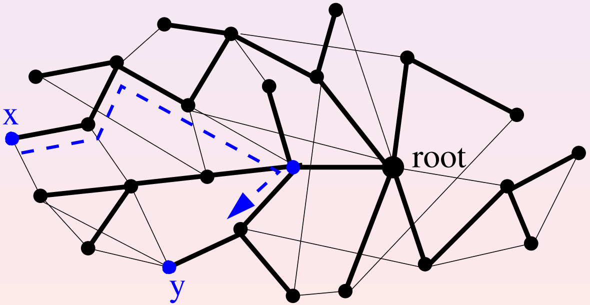


Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.

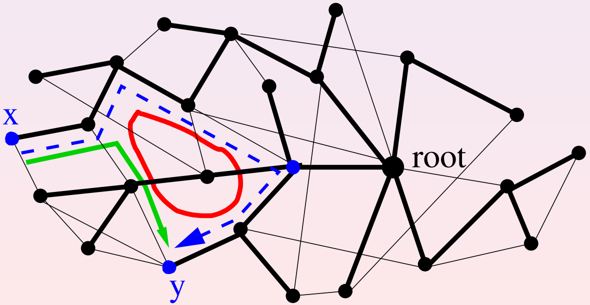


Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.



Simple Routing scheme [Nisse,Rapaport,Suchan 09]

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
    choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.

Once T has been chosen

Space: *labeling of nodes*: any rooted subtree $\Leftrightarrow$ interval
routing table: each node knows the interval of its neighbors
$O(\Delta \log n)$ bits per node

Time: easy to compute in time $O(D)$ in synchronous distributed way

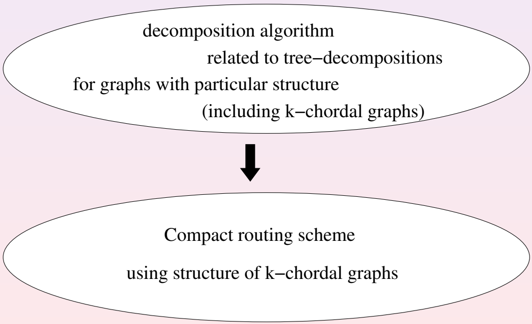
Stretch: if T is a BFS-tree in a k-chordal graph: additive stretch $\leq k - 1$

From Cops and robber to Routing via Treewidth

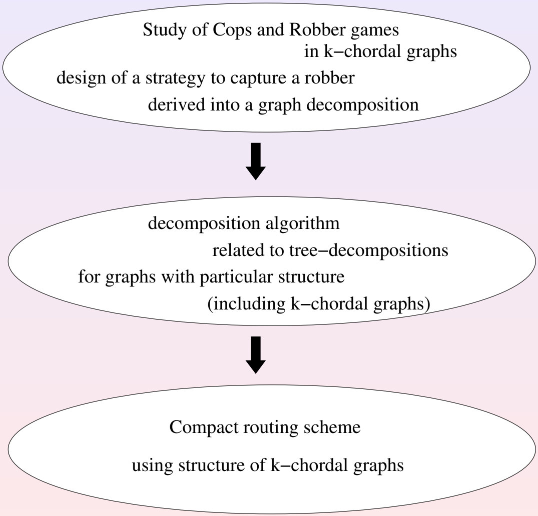
Compact routing scheme

using structure of k−chordal graphs

From Cops and robber to Routing via Treewidth



From Cops and robber to Routing via Treewidth



Our results [KLNS12]

Theorem 1: Cops and Robber games

$k-1$ cops are sufficient to capture a robber in k-chordal graphs

Theorem 2: main result

There is a $O(m^2)$-algorithm that, in any m-edge graph G,

- either returns an induced cycle larger than k,
- or compute a tree-decomposition with each bag being the closed neighborhood of an induced path of length $\leq k-1$.

($\Rightarrow$ treewidth $\leq O(\Delta.k)$ and treelength $\leq k$)

Theorem 3: for any graph admitting such a tree-decomposition

there is a compact routing scheme using RT's of size $O(k \log n)$ bits, and achieving additive stretch $O(k \log \Delta)$.

Outline

1. Distributed and Compact Routing
   - Routing in a distributed way
   - Compact Routing
   - Using structural Properties: chordality
   - Our results
2. Cops and Robber Games
   - Definitions and related works
   - Cops and Robber in k-chordal graphs
3. Structural result and link with compact routing

Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

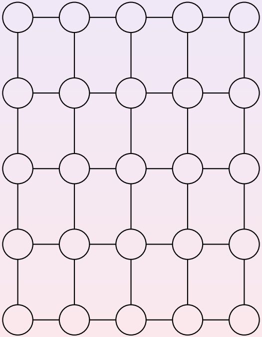
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

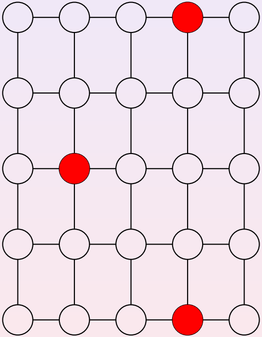
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

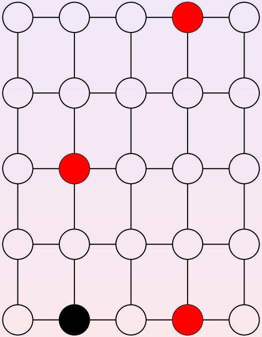
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

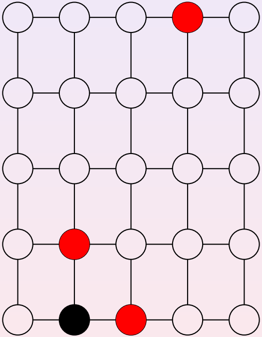
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

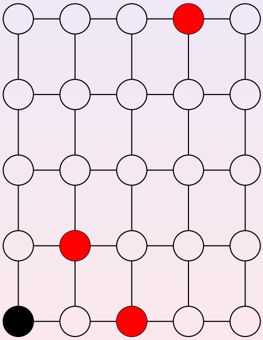
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

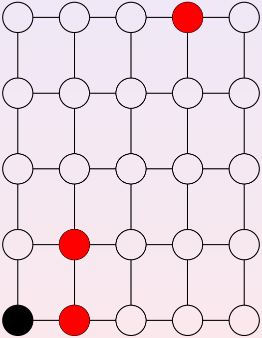
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

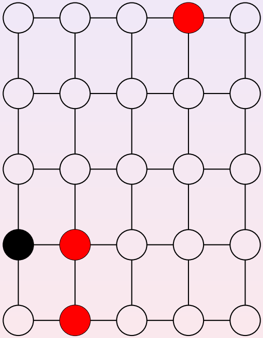
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

Robber captured:
A cop occupies the same vertex as the robber.



Cops & robber games [Nowakowski and Winkler; Quilliot, 83]

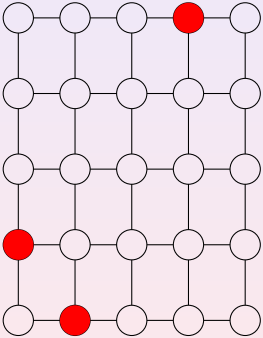
Initialization:

1. $\mathcal{C}$ places the cops;
2. $\mathcal{R}$ places the robber.

Step-by-step:

- each cop traverses at most 1 edge;
- the robber traverses at most 1 edge.

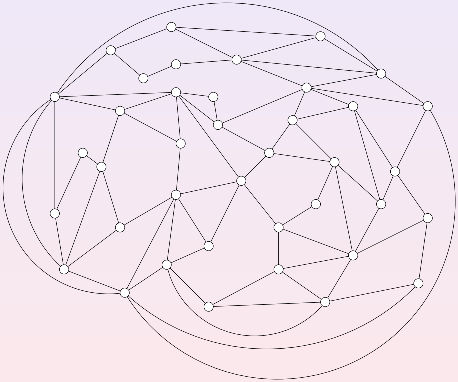
Robber captured:
A cop occupies the same vertex as the robber.



Cop number

$cn(G)$ minimum number of cops to capture any robber

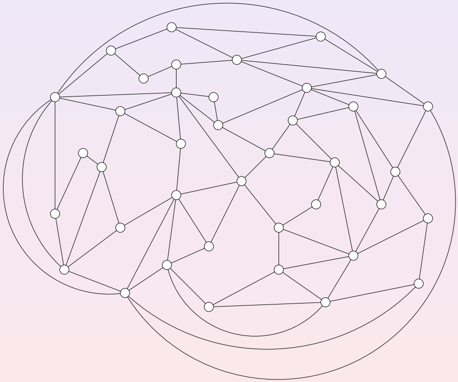
Determine $cn(G)$ for the following graph G?



Cop number

$cn(G)$ minimum number of cops to capture any robber

Determine $cn(G)$ for the following graph G? ≤ 3



$cn(G) \leq 3$ for any planar graph G [Aigner, Fromme, 84]

Computing $cn(G)$ is NP-hard [FGKNS 10]

Cops & robber games: the graph structure helps!!

- G with **girth** g (min induced cycle) and **min degree** d: $cn(G) \geq d^g$ [Frankl 87]
- $\exists$ n-node graphs G (projective plane): $cn(G) = \Theta(\sqrt{n})$ [Frankl 87]
- G with **dominating set** k: $cn(G) \leq k$ [folklore]
- **Planar graph** G: $cn(G) \leq 3$ [Aigner, Fromme, 84]
- **Minor free graph** G excluding a minor H: $cn(G) \leq |E(H)|$ [Andreae, 86]
- G with **genus** g: $cn(G) \leq 3/2g + 3$ [Schröder, 01]
- G with **treewidth** t: $cn(G) \leq t/2 + 1$ [Joret, Kaminsk,Theis 09]
- G **random graph** (Erdös Reyni): $cn(G) = O(\sqrt{n})$ [Bollobas *et al.* 08]
- **any** n-node graph G: $cn(G) = O(\frac{n}{2^{(1+o(1))\sqrt{\log n}}})$ [Lu,Peng 09, Scott,Sudakov 10]

Conjecture: For any connected n-node graph G, $cn(G) = O(\sqrt{n})$. [Meyniel 87]

Cops & robber games: the graph structure helps!!

- G with **girth** g (min induced cycle) and **min degree** d: $cn(G) \geq d^g$ [Frankl 87]
- $\exists$ n-node graphs G (projective plane): $cn(G) = \Theta(\sqrt{n})$ [Frankl 87]
- G with **dominating set** k: $cn(G) \leq k$ [folklore]
- **Planar graph** G: $cn(G) \leq 3$ [Aigner, Fromme, 84]
- **Minor free graph** G excluding a minor H: $cn(G) \leq |E(H)|$ [Andreae, 86]
- G with **genus** g: $cn(G) \leq 3/2g + 3$ [Schröder, 01]
- G with **treewidth** t: $cn(G) \leq t/2 + 1$ [Joret, Kaminsk,Theis 09]
- G **random graph** (Erdös Reyni): $cn(G) = O(\sqrt{n})$ [Bollobas *et al.* 08]
- **any** n-node graph G: $cn(G) = O(\frac{n}{2^{(1+o(1))\sqrt{\log n}}})$ [Lu,Peng 09, Scott,Sudakov 10]

Conjecture: For any connected n-node graph G, $cn(G) = O(\sqrt{n})$. [Meyniel 87]

Theorem 1 [KLNS12]

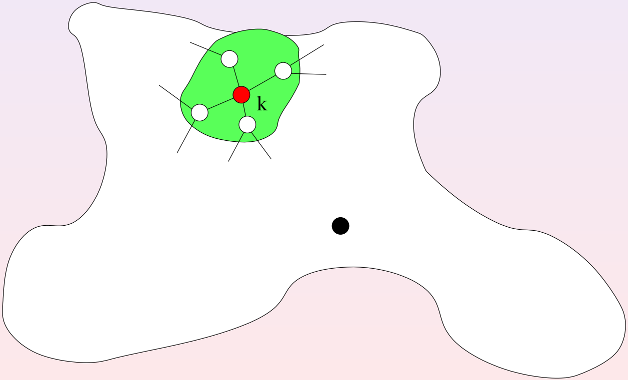
G with chordality k: $cn(G) \leq k - 1$.

Since 25 years, many researchers **study graphs structural properties** and introduce variants in the game to try solving the conjecture.
e.g., [Chiniforooshan 08, Bonato *et al.* 10, FGKNS 10, Alon,Mehrabian11, CCNV11, Clarke,McGillivray11]
see the recent survey book: The Game of Cops and Robbers on Graphs, A.Bonato and R.Nowakovski 2011

Caterpillar strategy — reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

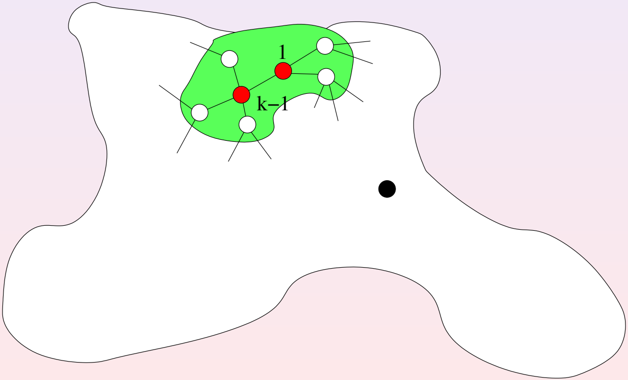
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

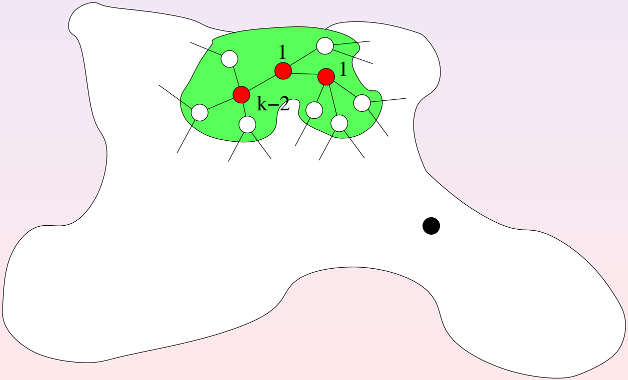
extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

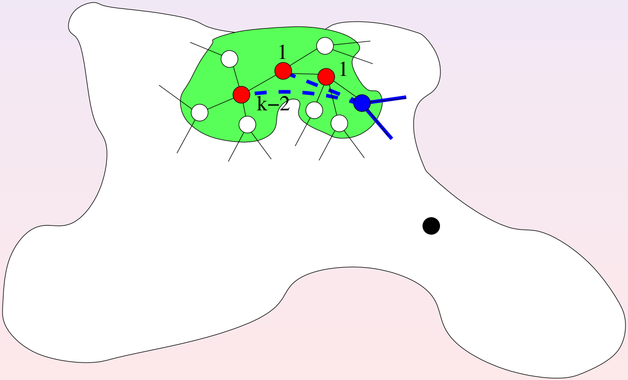
extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

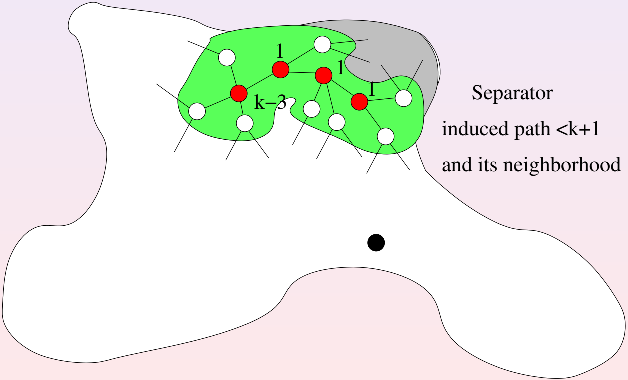
extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

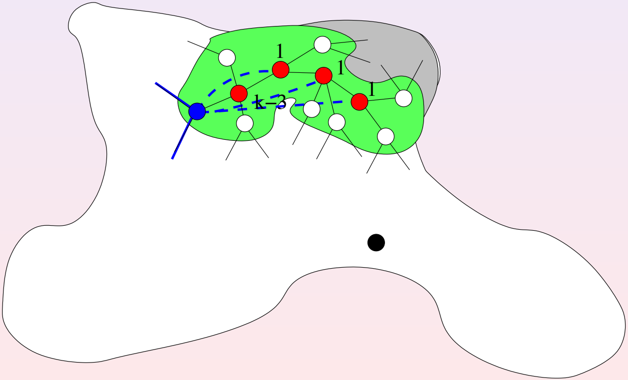
extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



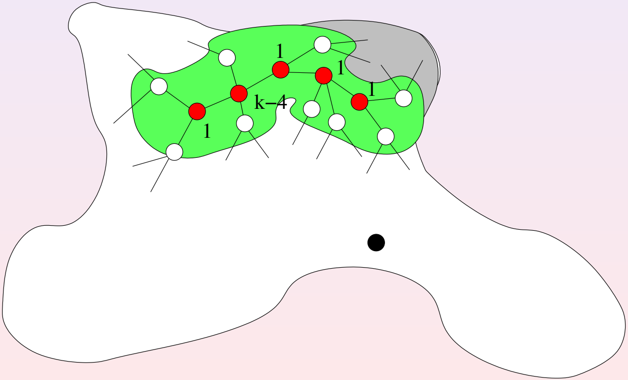
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

algorithm:

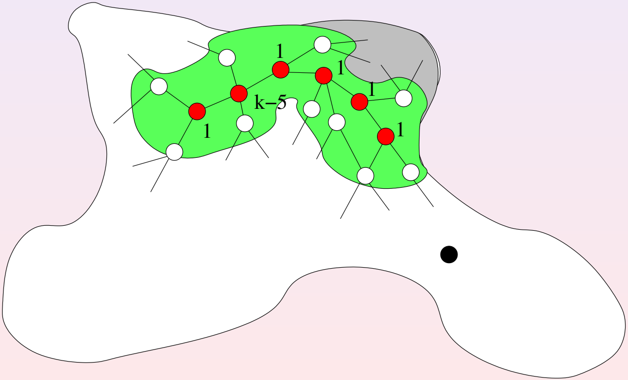
extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$
invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$
algorithm:

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



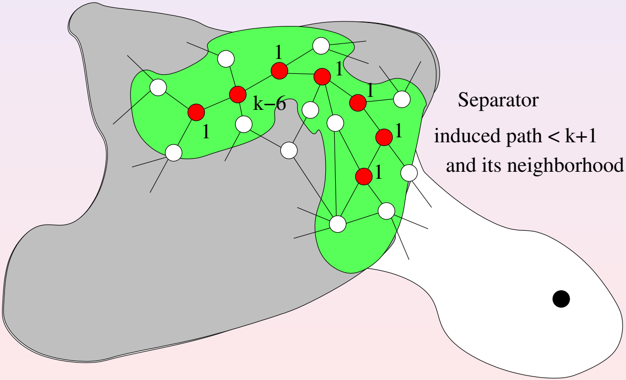
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

algorithm:

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



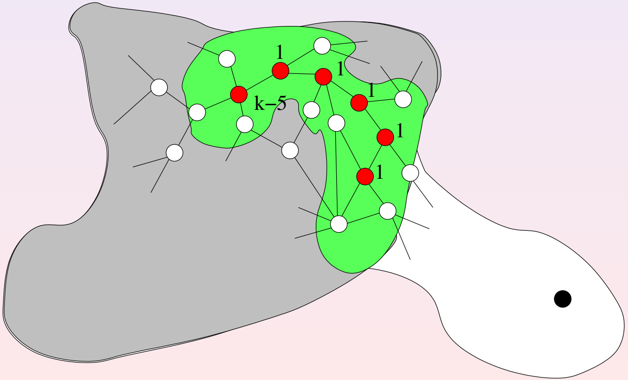
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

algorithm: *retraction:* if v_1 or v_i cannot be extended, else

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



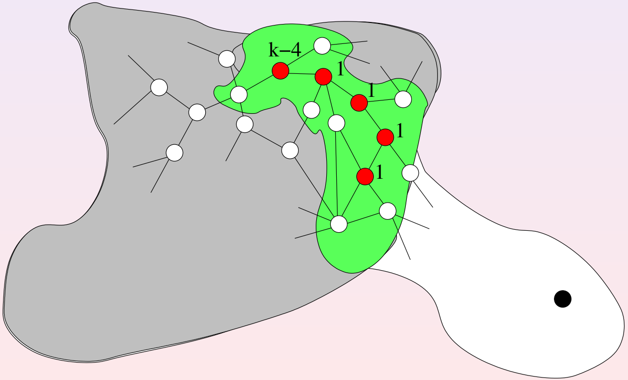
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

algorithm: *retraction:* if v_1 or v_i cannot be extended, else

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



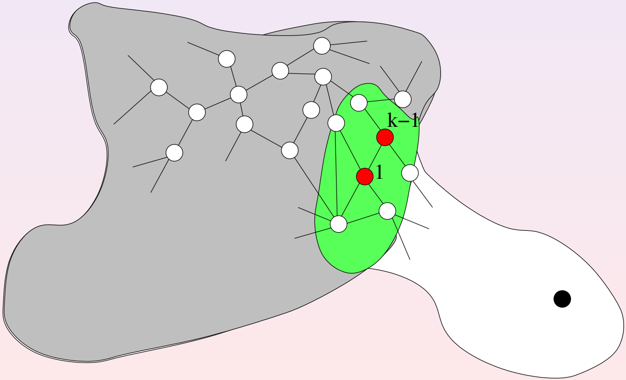
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

algorithm: *retraction:* if v_1 or v_i cannot be extended, else

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



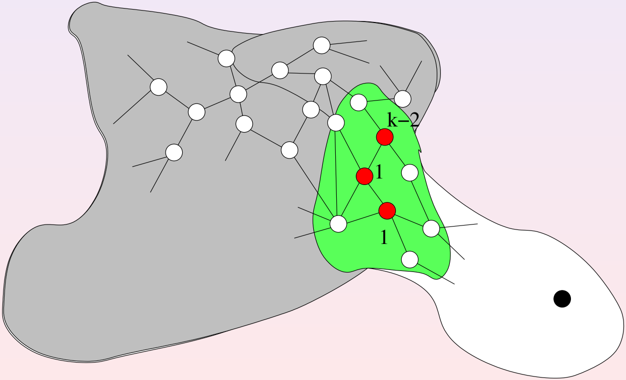
Caterpillar strategy reduce the robber area

initialization: all k cops in one arbitrary node $P = \{v_1\}$

invariant: Cops always occupy an induced path $P = \{v_1, \cdots, v_i\}$

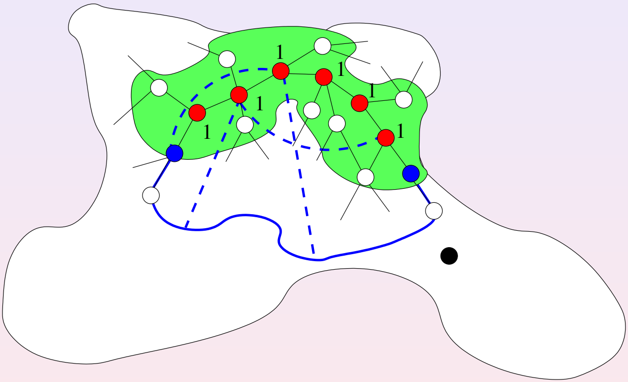
algorithm: *retraction:* if v_1 or v_i cannot be extended, else

extension: if $w \in N(v_1) \cup N(v_i)$, Pw induced and $N(w) \cap C_{robber} \neq \emptyset$



Capture in k-chordal graphs: Caterpillar strategy

$\{v_1, \cdots, v_i\}$ occupied: if no retraction $\Rightarrow$ induced cycle $\geq i+1$



Theorem 1 greedy algorithm

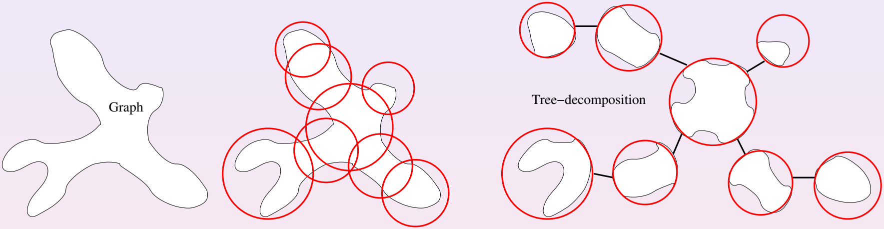
caterpillar strategy uses $\leq k-1$ cops in k-chordal graphs

Outline

1. Distributed and Compact Routing
   - Routing in a distributed way
   - Compact Routing
   - Using structural Properties: chordality
   - Our results
2. Cops and Robber Games
   - Definitions and related works
   - Cops and Robber in k-chordal graphs
3. Structural result and link with compact routing

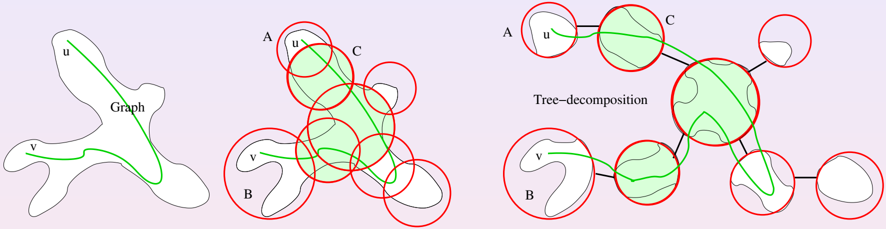
Tree-decomposition/treewidth (unformal)

Pieces (subgraphs) with tree-like structure (bag=separator)



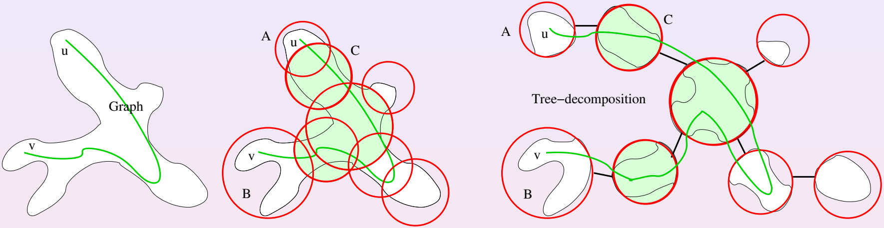
Tree-decomposition/treewidth (unformal)

Pieces (subgraphs) with tree-like structure (bag=separator)

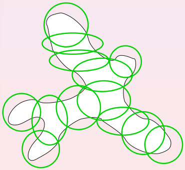


Tree-decomposition/treewidth (unformal)

Pieces (subgraphs) with tree-like structure (bag=separator)

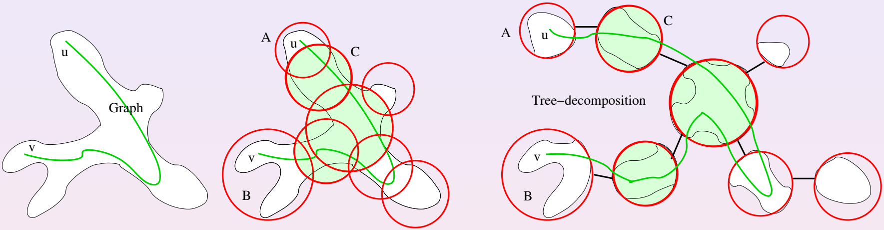


Usually, try to minimize the largest bag (treewidth)

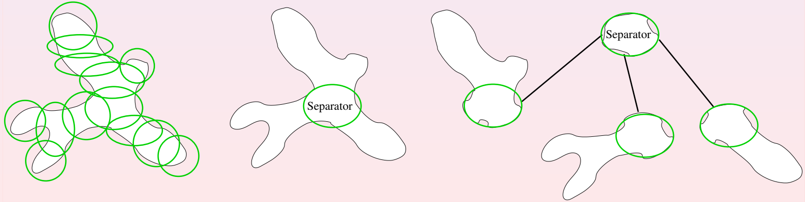


Tree-decomposition/treewidth (unformal)

Pieces (subgraphs) with tree-like structure (bag=separator)

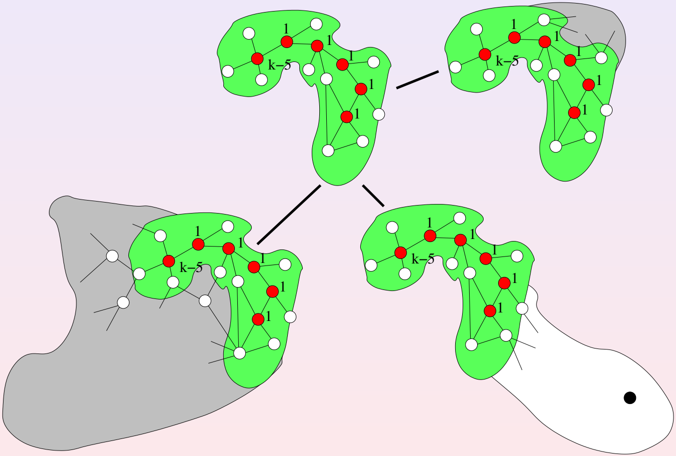


Computation: find a separator with desired properties, then induction



Tree-decomposition with k-induced paths

From k-Caterpillar's strategy



Tree-decomposition with k-induced paths

k-Caterpillar strategy $\Rightarrow$ decomposition with separator$=$ k-caterpillar

Theorem 2: main result [KLNS12]

There is a $O(m^2)$-algorithm that, in any m-edge graph G,

- either returns an induced cycle larger than k,
- or compute a tree-decomposition with each bag being the closed neighborhood of an induced path of length $\leq k-1$.

In case of k-chordal graphs:

$\Rightarrow$ treewidth $\leq O(\Delta.k)$ (improves [Bodlaender,Thilikos'97] result)

$\Rightarrow$ treelength $\leq k$ (already known)

$\Rightarrow$ hyperbolicity $\leq 3k/2$ (better bound is known)

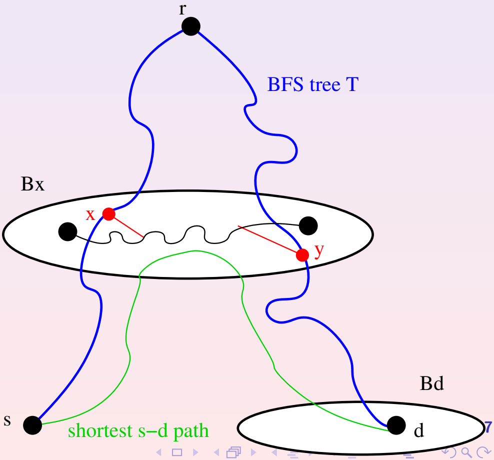
Application to compact routing

stretch $O(k \log \Delta)$ with RT's of size $O(k \log n)$ bits [KLNS12]

BFS-tree T, tree-decomposition D with k-caterpillar separators

From s to d

1. follow the path to r in T
   until find x such that
   B_x is an ancestor of B_d in D
   stretch: $+k$
2. in B_x, find y an ancestor of d in T
   stretch: $+k \log \Delta$
3. follow the path to d in T
   stretch: $+k$



Further Works

short term goals:

Implement the decomposition-algorithm and simulate on large scale models

What if other properties are included?

and then?

Structural Properties of Large-scale Networks

Efficient algorithms for $\geq 10^5$-node graphs

(e.g., for hyperbolicity $O(n^4)$-algorithms cannot work)
what about NP-hard problems? (e.g., chordality)
other "hidden" properties?

Distributed algorithms

How facing dynamicity?

models of dynamicity, localized algorithms

Cops and robber

Conjecture: For any connected n-node graph G, $cn(G) = O(\sqrt{n})$. [Meyniel 87]

Further Works

short term goals:

Implement the decomposition-algorithm and simulate on large scale models

What if other properties are included?

and then?

Structural Properties of Large-scale Networks

Efficient algorithms for $\geq 10^5$-node graphs

(e.g., for hyperbolicity $O(n^4)$-algorithms cannot work)

what about NP-hard problems? (e.g., chordality)

other "hidden" properties?

Distributed algorithms

How facing dynamicity?

models of dynamicity, localized algorithms

Cops and robber

Conjecture: For any connected n-node graph G, $cn(G) = O(\sqrt{n})$. [Meyniel 87]