On Robust Temporal Structures in Highly Dynamic Networks

Arnaud Casteigts
(LaBRI, University of Bordeaux)

J. work with Swan Dubois, Franck Petit, and John Michael Robson

https://arxiv.org/abs/1703.03190

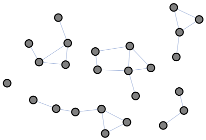
AATG@ICALP 2018

Highly dynamic networks

Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network

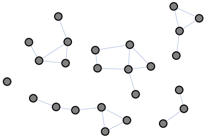


Highly dynamic networks

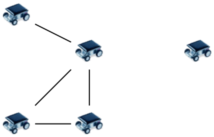
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



Example of scenario

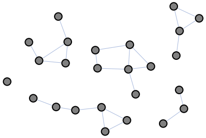


Highly dynamic networks

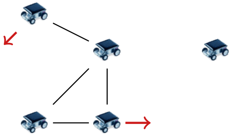
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



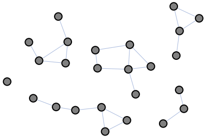
Example of scenario



Highly dynamic networks

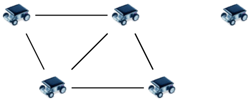
Ex: 

How changes are perceived?



- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network

Example of scenario

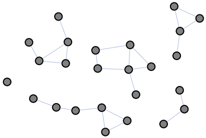


Highly dynamic networks

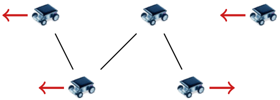
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



Example of scenario

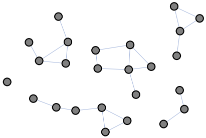


Highly dynamic networks

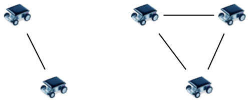
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



Example of scenario

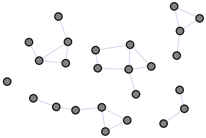


Highly dynamic networks

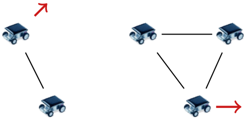
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



Example of scenario

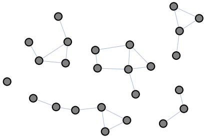


Highly dynamic networks

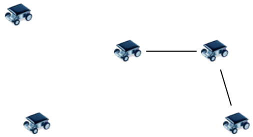
Ex: 

How changes are perceived?

- ~~Faults and Failures?~~
- Nature of the system. Change is normal.
- Possibly partitioned network



Example of scenario

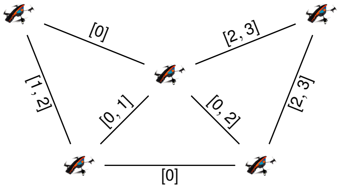


Graph representations

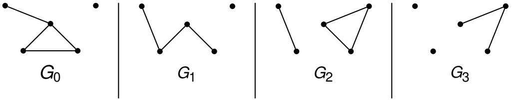
Time-varying graphs (TVG)

$\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$

- $\mathcal{T} \subseteq \mathbb{N}/\mathbb{R}$ (lifetime)
- $\rho : E \times \mathcal{T} \rightarrow \{0, 1\}$ (presence fonction)
- $\zeta : E \times \mathcal{T} \rightarrow \mathbb{N}/\mathbb{R}$ (latency function)



Another classical view $\mathcal{G} = G_0, G_1, \ldots$



Variety of models and terminologies:

Dynamic graphs, evolving graphs, temporal graphs, link streams, etc.

C., Flocchini, Quattrociocchi, Int. J. of Parallel, Emergent and Distributed Systems, Vol. 27, Issue 5, 2012

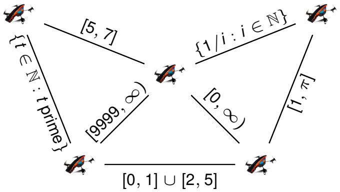
(among others)

Graph representations

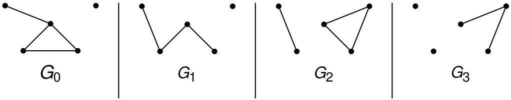
Time-varying graphs (TVG)

$\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$

- $\mathcal{T} \subseteq \mathbb{N}/\mathbb{R}$ (lifetime)
- $\rho : E \times \mathcal{T} \to \{0, 1\}$ (presence fonction)
- $\zeta : E \times \mathcal{T} \to \mathbb{N}/\mathbb{R}$ (latency function)



Another classical view $\mathcal{G} = G_0, G_1, \ldots$



Variety of models and terminologies:

Dynamic graphs, evolving graphs, temporal graphs, link streams, etc.

C., Flocchini, Quattrociocchi, Int. J. of Parallel, Emergent and Distributed Systems, Vol. 27, Issue 5, 2012

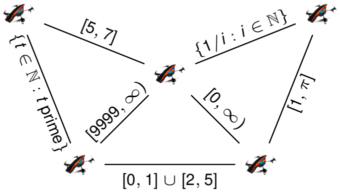
(among others)

Graph representations

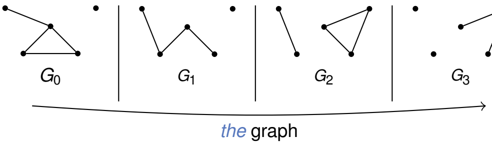
Time-varying graphs (TVG)

$\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$

- $\mathcal{T} \subseteq \mathbb{N}/\mathbb{R}$ (lifetime)
- $\rho : E \times \mathcal{T} \to \{0, 1\}$ (presence fonction)
- $\zeta : E \times \mathcal{T} \to \mathbb{N}/\mathbb{R}$ (latency function)



Another classical view $\mathcal{G} = G_0, G_1, \ldots$



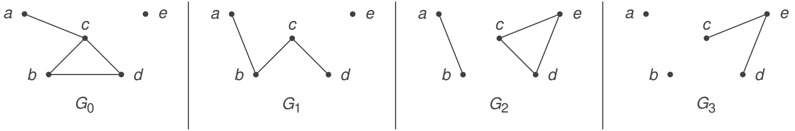
Variety of models and terminologies:

Dynamic graphs, evolving graphs, temporal graphs, link streams, etc.

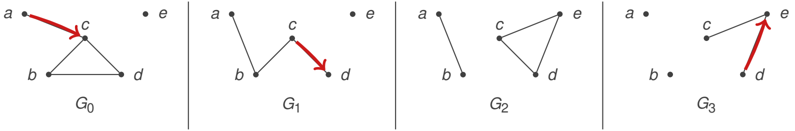
C., Flocchini, Quattrociocchi, Int. J. of Parallel, Emergent and Distributed Systems, Vol. 27, Issue 5, 2012

(among others)

Basic concepts



Basic concepts

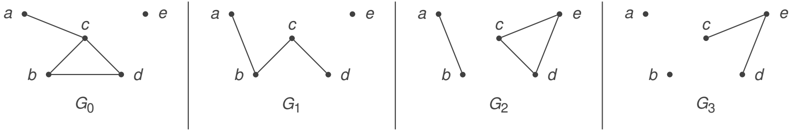


$\Longrightarrow$ *Temporal path* (*a.k.a. Journey*), *e.g.* $a \rightsquigarrow e$

Ex: $((ac, t_1), (cd, t_2), (de, t_3))$ with $t_{i+1} \geq t_i$ and $\rho(e_i, t_i) = 1$

(can be formulated with latency)

Basic concepts



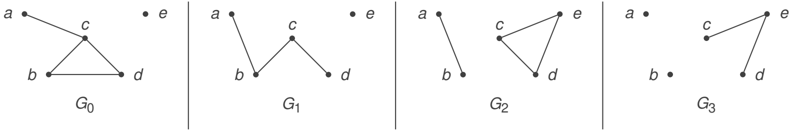
$\Longrightarrow$ *Temporal path* (*a.k.a. Journey*), *e.g.* $a \rightsquigarrow e$

Ex: $((ac, t_1), (cd, t_2), (de, t_3))$ with $t_{i+1} \geq t_i$ and $\rho(e_i, t_i) = 1$

(can be formulated with latency)

$\Longrightarrow$ *Temporal connectivity* $(* \rightsquigarrow *)$ Satisfied here?

Basic concepts



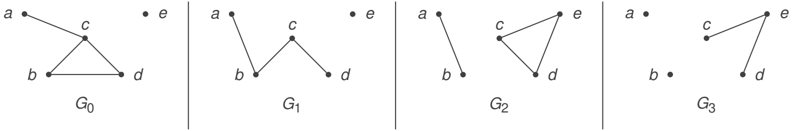
$\Longrightarrow$ *Temporal path* (*a.k.a. Journey*), *e.g.* $a \rightsquigarrow e$

Ex: $((ac, t_1), (cd, t_2), (de, t_3))$ with $t_{i+1} \geq t_i$ and $\rho(e_i, t_i) = 1$

(can be formulated with latency)

$\Longrightarrow$ *Temporal connectivity* $(* \rightsquigarrow *)$ Satisfied here? No, only $1 \rightsquigarrow *$.

Basic concepts



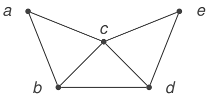
$\Longrightarrow$ *Temporal path* (*a.k.a. Journey*), *e.g.* $a \rightsquigarrow e$

Ex: $((ac, t_1), (cd, t_2), (de, t_3))$ with $t_{i+1} \geq t_i$ and $\rho(e_i, t_i) = 1$

(can be formulated with latency)

$\Longrightarrow$ *Temporal connectivity* $(* \rightsquigarrow *)$ Satisfied here? No, only $1 \rightsquigarrow *$.

$\Longrightarrow$ *Footprint ($\neq$ underlying graph)*

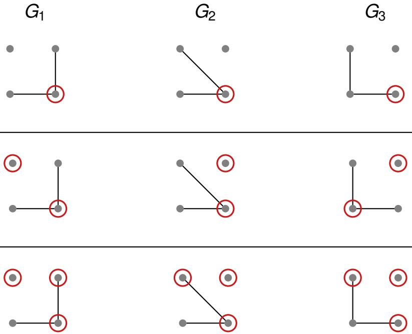


Today: Covering problems

Three ways of redefining covering problems

C., Mans, Mathieson, 2011

Ex: DOMINATINGSET



Temporal dominating set

Evolving dominating set

Permanent dominating set

Today: Covering problems

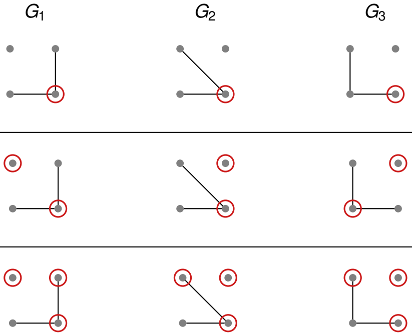
Three ways of redefining covering problems — C., Mans, Mathieson, 2011

Ex: DOMINATINGSET

Temporal dominating set

Evolving dominating set

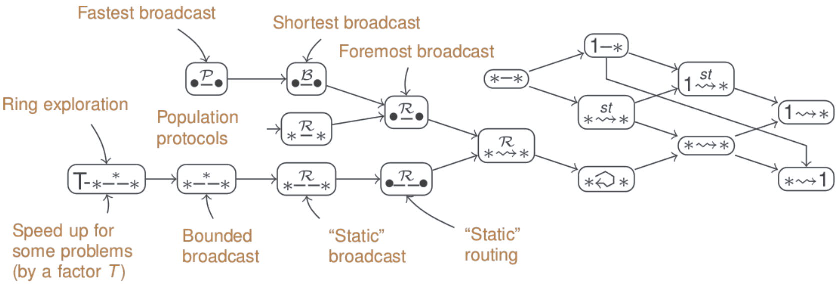
Permanent dominating set



$\rightarrow$ How about infinite time? The relation must hold infinitely often!

Classes of dynamic networks (C.,Flocchini,Quattrociocchi,Santoro, 2012)

What assumption for what problem?

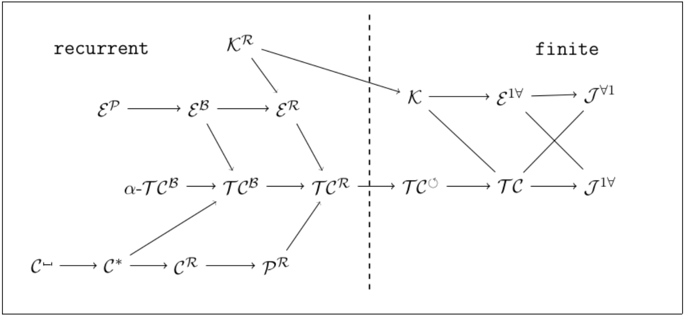


(based on time-varying graphs)

Classes of dynamic networks

(C.,Flocchini,Quattrociocchi,Santoro, 2012)

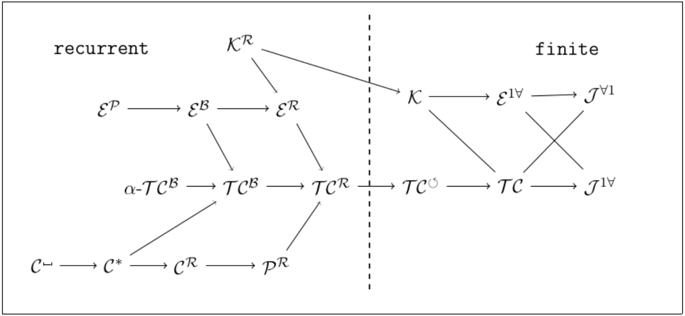
What assumption for what problem?



(C., 2018)

Classes of dynamic networks *(C.,Flocchini,Quattrociocchi,Santoro, 2012)*

What assumption for what problem?

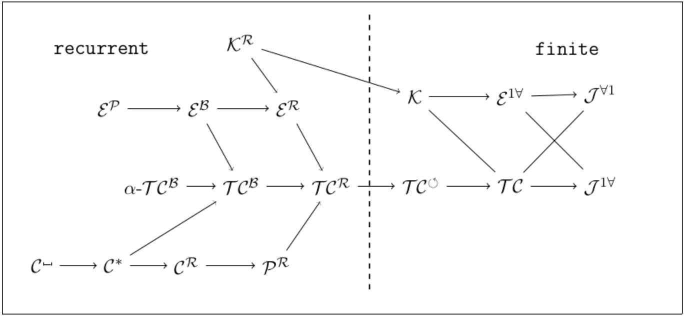


(C., 2018)

$\rightarrow \mathcal{E}^{\mathcal{R}} \equiv$ all the edges of the footprint are recurrent
$\rightarrow \mathcal{TC}^{\mathcal{R}} \equiv$ temporal connectivity is recurrently achived

Classes of dynamic networks *(C.,Flocchini,Quattrociocchi,Santoro, 2012)*

What assumption for what problem?



(C., 2018)

$\rightarrow$ $\mathcal{E}^{\mathcal{R}}$ $\equiv$ all the edges of the footprint are recurrent
$\rightarrow$ $\mathcal{TC}^{\mathcal{R}}$ $\equiv$ temporal connectivity is recurrently achived

Building temporal covering structures?

$\rightarrow$ $\mathcal{E}^{\mathcal{R}}$: “easy”
$\rightarrow$ $\mathcal{TC}^{\mathcal{R}}$: this talk

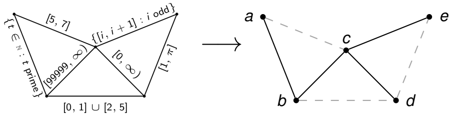
Exploiting regularities within $\mathcal{TC}^{\mathcal{R}}$

$\mathcal{TC}^{\mathcal{R}}$:= Temporal connectivity is recurrently achieved $(\mathcal{TC}^{\mathcal{R}} := \forall t, \mathcal{G}_{[t,+\infty)} \in \mathcal{TC})$

Exploiting regularities within $\mathcal{TC}^{\mathcal{R}}$

$\mathcal{TC}^{\mathcal{R}}$:= Temporal connectivity is recurrently achieved $\quad (\mathcal{TC}^{\mathcal{R}} := \forall t, \mathcal{G}_{[t,+\infty)} \in \mathcal{TC})$

Alternative characterization: $\mathcal{TC}^{\mathcal{R}} \equiv$ Eventual footprint connected — Braud Santoni et al., 2016

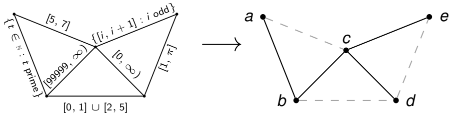


$\longrightarrow$ Can be exploited in a distributed algorithm — Kaaouachi *et al.*, 2016

Exploiting regularities within $\mathcal{TC}^{\mathcal{R}}$

$\mathcal{TC}^{\mathcal{R}}$:= Temporal connectivity is recurrently achieved $\quad (\mathcal{TC}^{\mathcal{R}} := \forall t, \mathcal{G}_{[t,+\infty)} \in \mathcal{TC})$

Alternative characterization: $\mathcal{TC}^{\mathcal{R}} \equiv$ Eventual footprint connected $\quad$ Braud Santoni et al., 2016



→ Can be exploited in a distributed algorithm $\quad$ Kaaouachi *et al.*, 2016

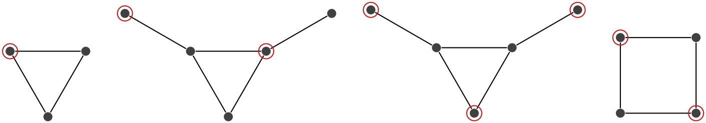
→ Robustness: New form of heredity asking that a property or solution holds in all connected spanning subgraphs

Ex: MinimalDominatingSet (MDS) and MaximalIndependentSet (MIS)

C., Dubois, Petit, Robson, 2017/18

Ex: Maximal Independent Sets

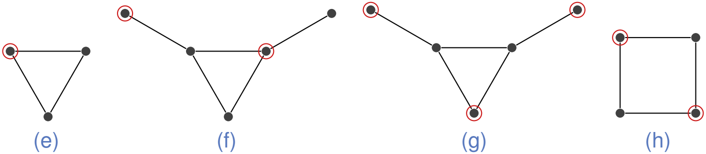
A maximal independent set (MIS) is a maximal ($\neq$ maximum) set of nodes, none of which are neighbors.



(a) (b) (c) (d)

EX: MAXIMAL INDEPENDENT SETS

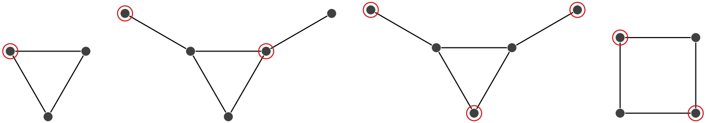
A maximal independent set (MIS) is a maximal ($\neq$ maximum) set of nodes, none of which are neighbors.



Which ones are robust?

Ex: Maximal Independent Sets

A maximal independent set (MIS) is a maximal ($\neq$ maximum) set of nodes, none of which are neighbors.



(i) (j) (k) (l)

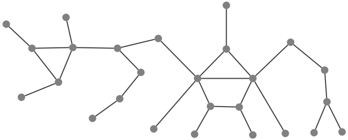
Which ones are robust?

$\rightarrow$ Question: characterizing graphs/footprints in which

1. all MISs are robust: ($\mathcal{RMIS}^{\forall}$)
2. at least one MIS is robust: ($\mathcal{RMIS}^{\exists}$)
3. all MDSs are robust: ($\mathcal{RMDS}^{\forall}$)
4. at least one MDS is robust: ($\mathcal{RMDS}^{\exists}$)

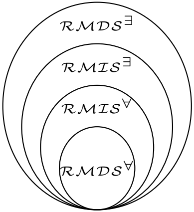
Overview of technical results

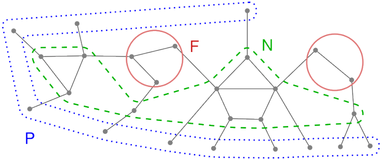
1. $\mathcal{RMDS}^\forall$ = Sputniks
2. $\mathcal{RMIS}^\forall$ = Complete bipartite $\cup$ Sputniks
3. $\mathcal{RMDS}^\exists \supsetneq$ bipartite + test algo
4. $\mathcal{RMIS}^\exists \supsetneq$ bipartite + test algo



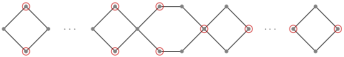
Locality:

1. $\mathcal{RMDS}^\forall$ and $\mathcal{RMIS}^\forall$
   $\rightarrow$ Robust solutions can be computed locally!
2. $\mathcal{RMIS}^\exists$
   $\rightarrow$ Robust solutions cannot be computed locally!





Local algo for robust MIS in Sputniks



Lower bound on the non-locality of robust MIS

$\mathcal{RMIS}^{\forall}$

Graphs in which *all* MISs are robust? ($\mathcal{RMIS}^{\forall}$)

$\mathcal{RMIS}^{\forall}$

Graphs in which *all* MISs are robust? ($\mathcal{RMIS}^{\forall}$)

Lemma

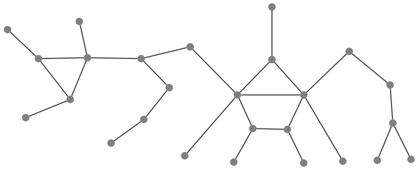
Bipartite complete ($\mathcal{BK}$) graphs $\subseteq \mathcal{RMIS}^{\forall}$.

$\mathcal{RMIS}^{\forall}$

Graphs in which *all* MISs are robust? ($\mathcal{RMIS}^{\forall}$)

Lemma

Bipartite complete ($\mathcal{BK}$) graphs $\subseteq \mathcal{RMIS}^{\forall}$.



Def: A graph is a sputnik if and only if every node that belongs to a cycle also has an antenna (*i.e.* a pendant neighbor).

Lemma

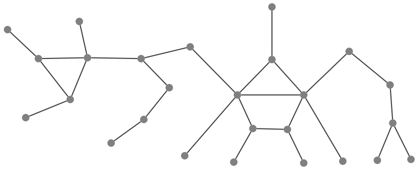
Sputniks $\subseteq \mathcal{RMIS}^{\forall}$.

$\mathcal{RMIS}^\forall$

Graphs in which *all* MISs are robust? ($\mathcal{RMIS}^\forall$)

Lemma

Bipartite complete ($\mathcal{BK}$) graphs $\subseteq \mathcal{RMIS}^\forall$.



Def: A graph is a sputnik if and only if every node that belongs to a cycle also has an antenna (*i.e.* a pendant neighbor).

Lemma

Sputniks $\subseteq \mathcal{RMIS}^\forall$.

Theorem

$\mathcal{RMIS}^\forall = \textit{Sputniks} \cup \mathcal{BK}$

Local algorithm to find a RMIS in $\mathcal{RMIS}^{\forall}$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log\log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log\log n)$ [BE10]

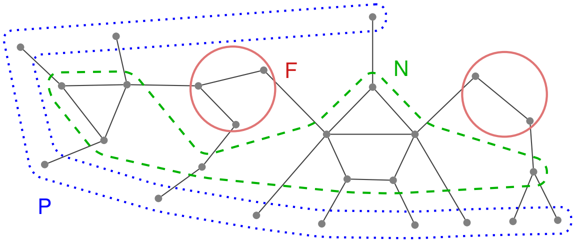
Can we solve the problem locally in $\mathcal{RMIS}^{\forall}$?

Local algorithm to find a RMIS in $\mathcal{RMIS}^{\forall}$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log\log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log\log n)$ [BE10]

Can we solve the problem locally in $\mathcal{RMIS}^{\forall}$?



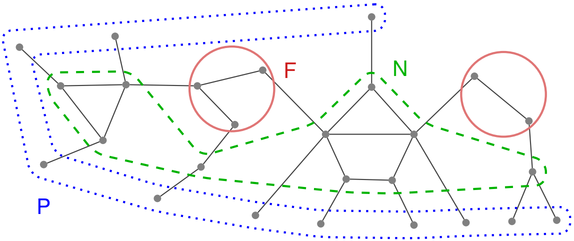
P: pendant node
N: neighbor of a pendant node
F: other

Local algorithm to find a RMIS in $\mathcal{RMIS}^{\forall}$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log \log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log \log n)$ [BE10]

Can we solve the problem locally in $\mathcal{RMIS}^{\forall}$?



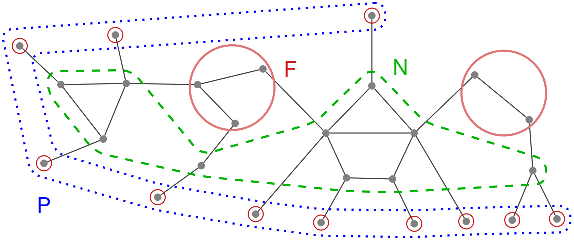
P: pendant node
N: neighbor of a pendant node
F: other

Local algorithm to find a RMIS in $\mathcal{RMIS}^{\forall}$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log\log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log\log n)$ [BE10]

Can we solve the problem locally in $\mathcal{RMIS}^{\forall}$?



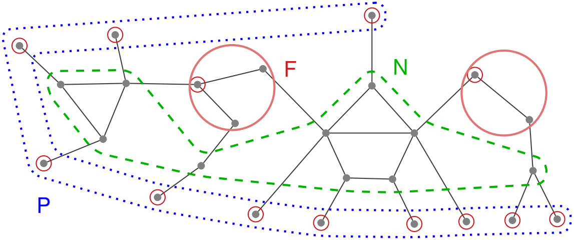
P: pendant node
N: neighbor of a pendant node
F: other

Local algorithm to find a RMIS in $\mathcal{RMIS}^{\forall}$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log\log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log\log n)$ [BE10]

Can we solve the problem locally in $\mathcal{RMIS}^{\forall}$?



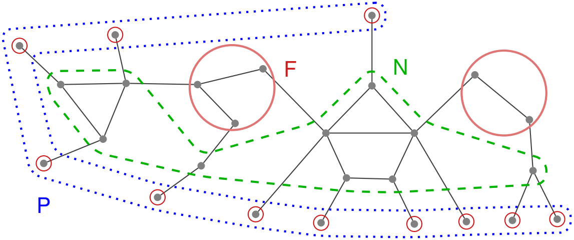
P: pendant node
N: neighbor of a pendant node
F: other

Local algorithm to find a RMIS in $\mathcal{RMIS}^\forall$

State of the art (classical MIS)

- Lower bound: $\Omega(\sqrt{\log n / \log\log n})$ [KMW04]
- Best algo: $2^{O(\sqrt{\log n})}$ [PS96] (between $\log n$ and n)
- Best algo in trees: $O(\log n / \log\log n)$ [BE10]

Can we solve the problem locally in $\mathcal{RMIS}^\forall$?



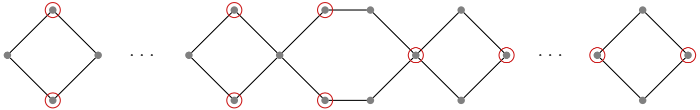
P: pendant node
N: neighbor of a pendant node
F: other

$\Longrightarrow$ $\boxed{o(\log n)}$

Not local in general graphs! (i.e. $\Omega(n)$)

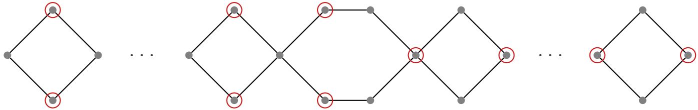
Not local in general graphs! (i.e. $\Omega(n)$)

$\exists$ Infinite family of graphs $(G_k)_{k \in \mathbb{N}}$, of diameter $\Theta(k) = \Theta(n)$.



Not local in general graphs! (i.e. $\Omega(n)$)

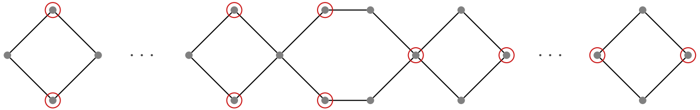
$\exists$ Infinite family of graphs $(G_k)_{k\in\mathbb{N}}$, of diameter $\Theta(k) = \Theta(n)$.



<u>Lemma:</u> $\forall k$, G_k admits only two robust MISs M_1 (in red) and $M_2 = V \setminus M_1$.

Not local in general graphs! (i.e. $\Omega(n)$)

$\exists$ Infinite family of graphs $(G_k)_{k\in\mathbb{N}}$, of diameter $\Theta(k) = \Theta(n)$.

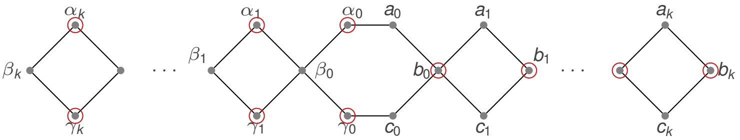


Lemma: $\forall k$, G_k admits only two robust MISs M_1 (in red) and $M_2 = V \setminus M_1$.

(1) Anonymous case (easy): Both extremities have same view up to distance $\Theta(n)$,
but they must decide differently. $\square$

Not local in general graphs! (i.e. $\Omega(n)$)

$\exists$ Infinite family of graphs $(G_k)_{k \in \mathbb{N}}$, of diameter $\Theta(k) = \Theta(n)$.



Lemma: $\forall k$, G_k admits only two robust MISs M_1 (in red) and $M_2 = V \setminus M_1$.

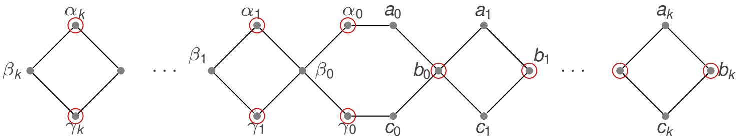
(1) Anonymous case (easy): Both extremities have same view up to distance $\Theta(n)$, but they must decide differently. $\square$

(2) Identified networks: let $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ be disjoint labeling functions that assign identifiers to $n/3$ nodes starting at one extremity (left or right). Let the whole graph be labeled either (1) $\mathcal{L}_1 \cdot x \cdot \mathcal{L}_2$; (2) $\mathcal{L}_1 \cdot y \cdot \mathcal{L}_3$; (3) $\mathcal{L}_2 \cdot z \cdot \mathcal{L}_3$, with x, y, and z arbitrary.

Unless using information within $\Omega(n)$ hops, β_k and b_k will decide identically in some cases, whatever the algorithm. $\square$

Not local in general graphs! (i.e. $\Omega(n)$)

$\exists$ Infinite family of graphs $(G_k)_{k\in\mathbb{N}}$, of diameter $\Theta(k) = \Theta(n)$.



Lemma: $\forall k$, G_k admits only two robust MISs M_1 (in red) and $M_2 = V \setminus M_1$.

(1) Anonymous case (easy): Both extremities have same view up to distance $\Theta(n)$,
but they must decide differently. $\square$

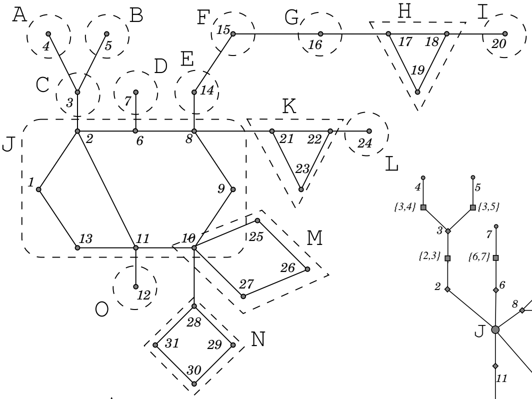
(2) Identified networks: let $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ be disjoint labeling functions that assign identifiers to $n/3$ nodes starting at one extremity (left or right). Let the whole graph be labeled either (1) $\mathcal{L}_1 \cdot x \cdot \mathcal{L}_2$; (2) $\mathcal{L}_1 \cdot y \cdot \mathcal{L}_3$; (3) $\mathcal{L}_2 \cdot z \cdot \mathcal{L}_3$, with x, y, and z arbitrary.

Unless using information within $\Omega(n)$ hops, β_k and b_k will decide identically in some cases, whatever the algorithm. $\square$

$\rightarrow$ Essentially as bad as collecting all information at one node and use offline algo.

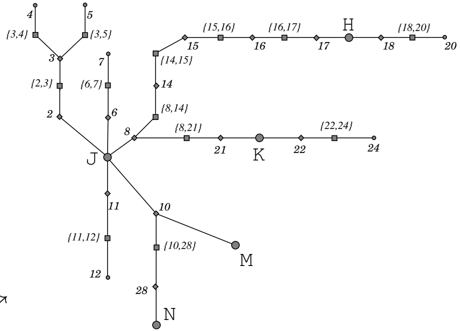
Centralized algorithm to find RMISs in general (in **P**)

Objective: Finds a RMIS if one exists, rejects otherwise.

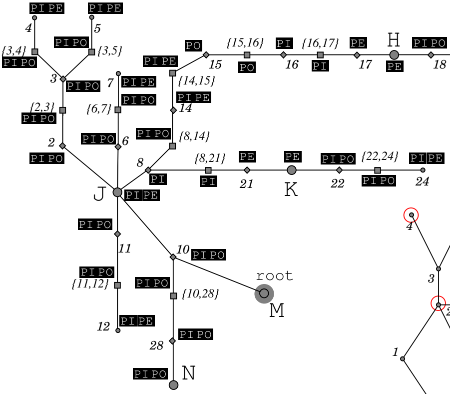


↑
Decomposition into biconnected components

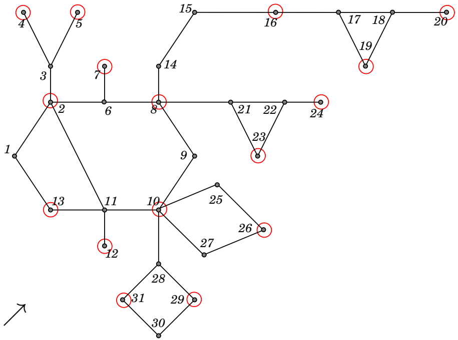
$\mathcal{ABC}$-tree ↗



Polynomial-time algorithm to find RMISs (2)



↑
Tagging



Resulting RMIS ↗

Děkuji !