Finding and Exploiting Structure in Highly-Dynamic Networks

Arnaud Casteigts

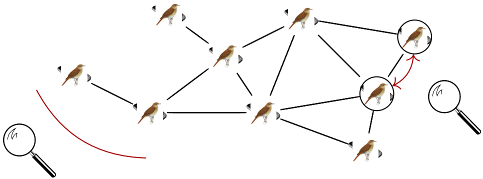
Rescom Summer School 2019 (Anglet)

June 24, 2019

Networks?

- Set of nodes V (*a.k.a.* entities, vertices)
- Set of links E among them (*a.k.a.* relations, edges)

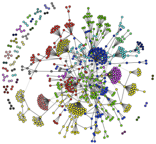
$\rightarrow$ A network (or graph) $G = (V, E)$



Complex networks (data analysis)

$\rightarrow$ detect patterns

$\rightarrow$ explain and reproduce phenomena



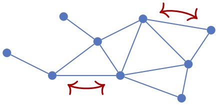
Communication networks

$\rightarrow$ design interactions among entities

$\rightarrow$ study what can be done *from within*

$\rightarrow$ distributed algorithms...

Distributed Algorithms



Collaboration of distinct entities to perform a common task.

No centralization available. No global knowledge.

(Think globally, act locally)

Examples of problems

Broadcast

Propagating a piece of information from one node to all others.

Election

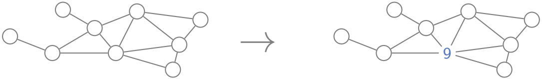
Distinguishing exactly one node among all.

Spanning tree

Selecting a cycle-free set of edges that interconnects all nodes.

Counting

Determining how many participants there are.

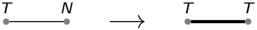


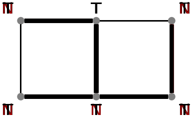
Consensus, naming, routing, exploration, ...

Abstracting Communications

Atomic interaction

(Population protocols *(Angluin et al., 2004)*; Graph relabeling systems *(Litovsky et al., 1999)*)

Ex: 



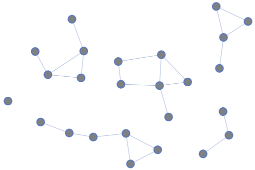
Note: Scheduling is not part of the algorithm!

→ Can be adversarial, randomized, etc.

Scope of the models

Relations between them *(Chalopin, 2006)*

Dynamic Networks



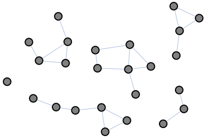
Dynamic networks?

In fact, *highly* dynamic networks.

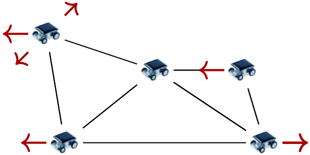
Ex: 

How changes are perceived?

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



Example of scenario
(say, exploration by mobile robots)

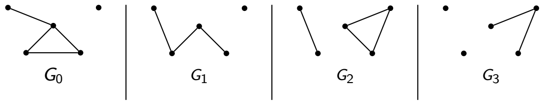


Dynamic Graphs

Also called *time-varying graphs*, *evolving graphs*, *temporal graphs*, *etc.*

As a sequence

Sequence of static graphs $\mathcal{G} = G_0, G_1, \ldots$ [+table of dates]

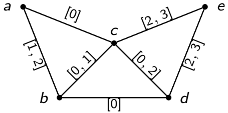


With a presence function

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

with ρ being a *presence function*

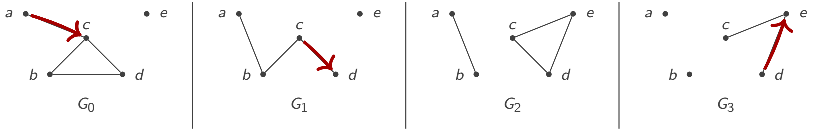
$\rho : E \times \mathcal{T} \to \{0, 1\}$



$\rightarrow$ These models (and others) are essentially equivalent if $\mathcal{T} \subseteq \mathbb{N}$
(not when $\mathcal{T} \subseteq \mathbb{R}$, e.g. )

$\rightarrow$ Further extensions possible (latency function, node-presence function, ...)

Basic graph concepts



→ Paths become temporal (*journey*)

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

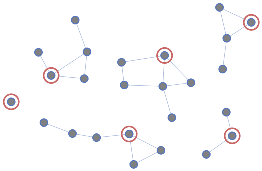
Also known as *Schedule-conforming path*, *Time-respecting path*, *Temporal path*, and *Journey* *(Bui-Xuan et al., 2003)*.

→ *Strict* journeys *vs.* *non-strict* journeys. (More relevant in discrete time.)

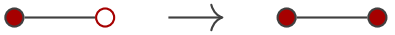
→ *Temporal connectivity*. Not symmetrical! (e.g. $a \rightsquigarrow e$, but $e \not\rightsquigarrow a$)

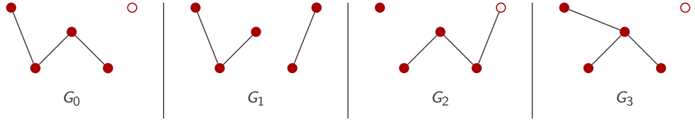
→ Snapshots, footprint, *etc.*

Necessary and sufficient conditions in dynamic networks



Informal example

Ex: Broadcast algorithm 



Lucky version. Yeah !! But things could have gone differently. Too late! Failure! Or even worse.. Too fast! Too fast! Failure!

$\Longrightarrow$ Additional assumptions needed to guarantee something.

Assumption: Every present edge is "selected" at least once (but we don't know in what order...)

$\rightarrow$ Now, is the success *guaranteed*? Why not? Because $\neg(src \overset{st}{\rightsquigarrow} *)$

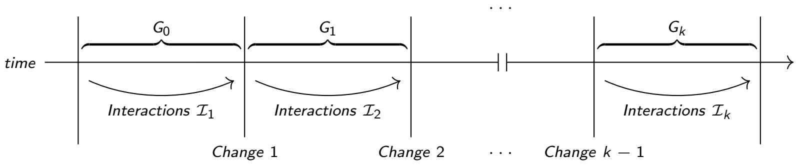
$\rightarrow$ Is the success *possible*? Of course, but why? Because $(src \rightsquigarrow *)$

Notions of *necessary condition* (e.g. $src \rightsquigarrow *$) or *sufficient condition* (e.g. $src \overset{st}{\rightsquigarrow} *$) for a given algorithm. These conditions relate only to the topology.

More formally...

Interaction over dynamic graphs

Interactions over a Dynamic Graph $\mathcal{G} = \{G_0, G_1, ..., G_k\}$



An execution is an alternated sequence of interactions and topological events:

$X = \mathcal{I}_k \circ Change_{k-1} \circ .. \circ Change_2 \circ \mathcal{I}_2 \circ Change_1 \circ \mathcal{I}_1(G_0)$ *Non deterministic !*

$\rightarrow$ $\mathcal{X}$: set of all possible executions (for a given algorithm and graph $\mathcal{G}$).

What makes a graph property $\mathcal{P}$ a necessary or sufficient condition for success on $\mathcal{G}$?

- $\rightarrow$ *Necessary condition*: $\neg\mathcal{P}(\mathcal{G}) \implies \forall X \in \mathcal{X}$, failure(X).
- $\rightarrow$ *Sufficient condition*: $\mathcal{P}(\mathcal{G}) \implies \forall X \in \mathcal{X}$, success(X).

Back to the broadcast example

Necessary condition

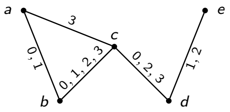
$\rightarrow$ $\mathcal{P}_{\mathcal{N}}$: there exists a journey from the source to all other nodes (noted $src \rightsquigarrow *$).

Sufficient condition

$\rightarrow$ $\mathcal{P}_{\mathcal{S}}$: there exists a *strict* journey from the source to all other nodes (noted $src \overset{st}{\rightsquigarrow} *$).

Classes of dynamic graphs

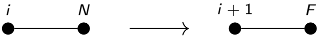
- $\rightarrow$ $\mathcal{C}_1$: $\mathcal{P}_{\mathcal{N}}$ is satisfied by at least one node (noted $1 \rightsquigarrow *$).
- $\rightarrow$ $\mathcal{C}_2$: $\mathcal{P}_{\mathcal{N}}$ is satisfied by all nodes ($* \rightsquigarrow *$).
- $\rightarrow$ $\mathcal{C}_3$: $\mathcal{P}_{\mathcal{S}}$ is satisfied by at least one node ($1 \overset{st}{\rightsquigarrow} *$).
- $\rightarrow$ $\mathcal{C}_4$: $\mathcal{P}_{\mathcal{S}}$ is satistied by all nodes ($* \overset{st}{\rightsquigarrow} *$).



$\in \mathcal{C}_2 \cap \mathcal{C}_3$

Counting algorithm (non-uniform)

Counting with a distinguished counter

- Initial states: 1 for the counter, N for all other nodes.
- Algorithm: 

$\rightarrow$ Hopefully, after some time, the counter is labelled n. But when?

Necessary or sufficient conditions

- $\mathcal{P}_{\mathcal{N}}$: there exists an edge, at some time, between the counter and every other node.
- $\mathcal{P}_{\mathcal{S}} = \mathcal{P}_{\mathcal{N}}$.

Classes of dynamic graphs

$\rightarrow$ $\mathcal{C}_5$: at least one node verifies $\mathcal{P}$, (noted 1–$*$).

$\rightarrow$ $\mathcal{C}_6$: all the nodes verify $\mathcal{P}$, (noted $*$–$*$).

Counting algorithm (uniform)

Uniform counting (every body is initially a counter)

- Initial states: 1 (all nodes).
- Algorithm: $\overset{i \neq 0}{\bullet}\!\!-\!\!\!-\!\!\!-\!\!\overset{j \neq 0}{\bullet} \longrightarrow \overset{i+j}{\bullet}\!\!-\!\!\!-\!\!\!-\!\!\overset{0}{\bullet}$

$\rightarrow$ Hopefully, after some time, one node is labelled n. But when?

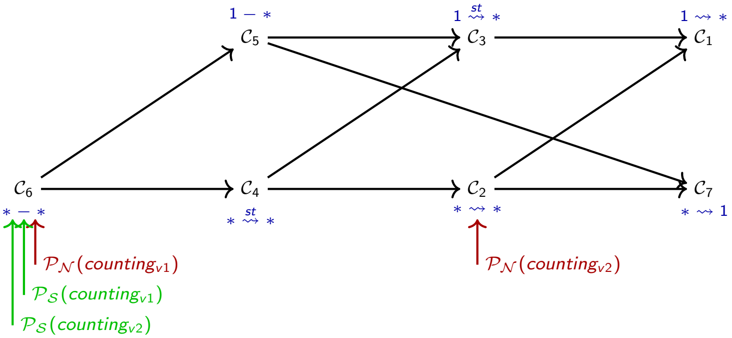
Conditions and classes of graphs

- Necessary condition $\mathcal{C}_{\mathcal{N}}$: at least one node can be reached by all ($* \rightsquigarrow 1$).
  - $\rightarrow$ $\mathcal{C}_7$: graphs having this property.
- Sufficient condition $\mathcal{C}_{\mathcal{S}}$: all pairs of nodes must share an edge at least once over time ($*{-}*$).
  - $\rightarrow$ $\mathcal{C}_6$ (already seen before).

Classifying dynamic networks

$$\begin{tikzcd}
 & \overset{1 - *}{\mathcal{C}_5} \arrow[rr, "- \implies \mathcal{J}_{strict}"] \arrow[rrrrd, "- \implies \mathcal{J}"] & & \overset{1 \overset{st}{\rightsquigarrow} *}{\mathcal{C}_3} \arrow[rr, "\mathcal{J}_{strict} \implies \mathcal{J}"] & & \overset{1 \rightsquigarrow *}{\mathcal{C}_1} \\
\underset{* - *}{\mathcal{C}_6} \arrow[ru, "\forall \implies \exists"] \arrow[rr, "- \implies \mathcal{J}_{strict}"'] & & \underset{* \overset{st}{\rightsquigarrow} *}{\mathcal{C}_4} \arrow[ru, "\forall \implies \exists"] \arrow[rr, "\mathcal{J}_{strict} \implies \mathcal{J}"'] & & \underset{* \rightsquigarrow *}{\mathcal{C}_2} \arrow[ru, "\forall \implies \exists"] \arrow[r, "\forall \implies \exists"'] & \underset{* \rightsquigarrow 1}{\mathcal{C}_7}
\end{tikzcd}$$

Classifying dynamic networks

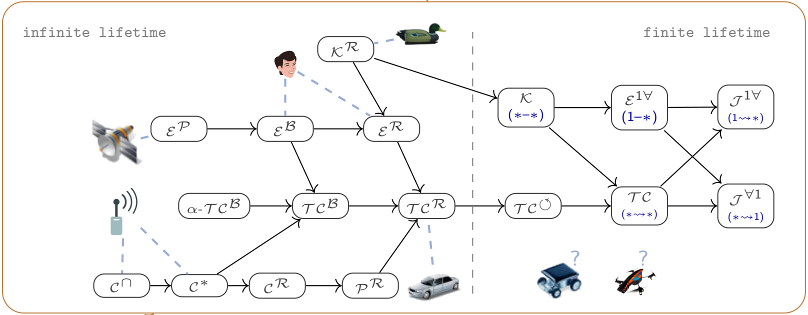


$\rightarrow$ Comparison of distributed algorithms on a formal basis

Further classes of dynamic networks

Network algorithms

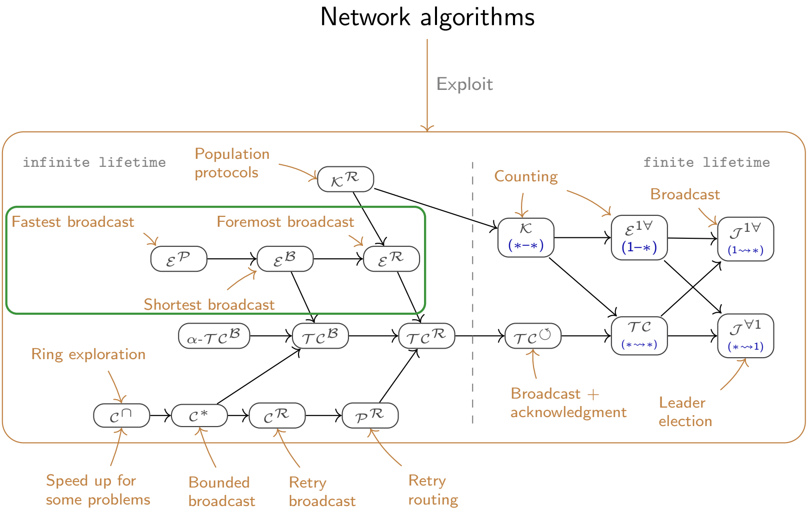
Exploit



Test

Data analysis

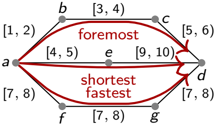
Classes of dynamic networks



Zoom: Optimal broadcast in DTNs?

What optimality ?

(Bui-Xuan, Jarry, Ferreira, 2003)



Which way is optimal from a to d?

-min hop? (shortest)

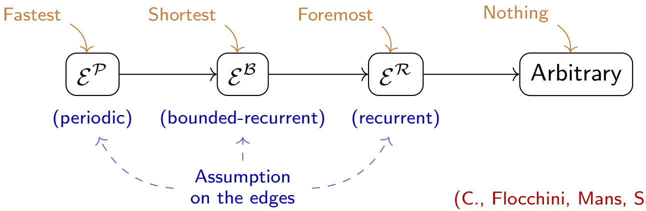
-earliest arrival? (foremost)

-fastest traversal? (fastest)

→ Computing shortest, foremost and fastest journeys in dynamic networks

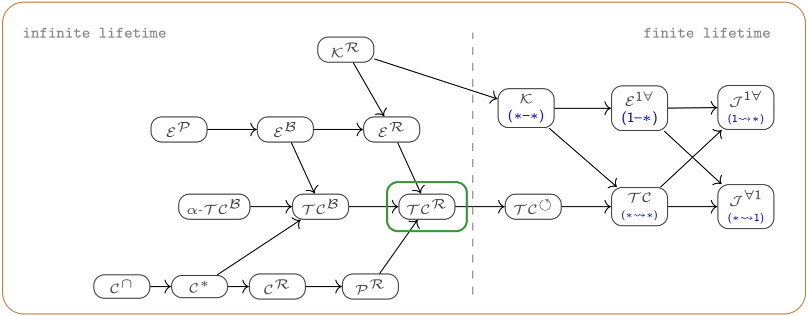
What about the distributed version?

→ Can we broadcast a message to all the nodes in a foremost, shortest, or fastest way? (with termination detection at the emitter and without knowing the schedule)



(C., Flocchini, Mans, Santoro, 2015)

Classes of dynamic networks

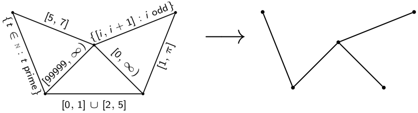


Zoom: Exploiting structure within $\mathcal{TC}^{\mathcal{R}}$

$\mathcal{TC}^{\mathcal{R}}$:= All nodes can reach each other through journeys infinitely often
(Formally, $\mathcal{TC}^{\mathcal{R}} := \forall t, \mathcal{G}_{[t,+\infty)} \in \mathcal{TC}$)

$\equiv$ A connected spanning subset of the edges must be recurrent (Braud Santoni et al., 2016)

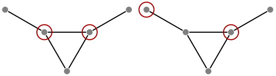
For example,



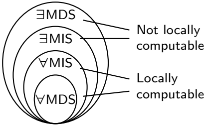
Can we exploit this property even if we don't know which subset of edges is recurrent?

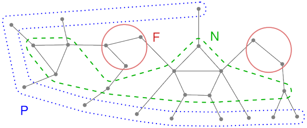
$\rightarrow$ Yes, e.g. for covering problems like MINIMALDOMINATINGSET, in some cases a solution can be found relative to the footprint and remain effective in all possible eventual footprints provided the graph is in $\mathcal{TC}^{\mathcal{R}}$. Such a solution is then called *robust* (C., Dubois, Petit, Robson, 2018)

Example:



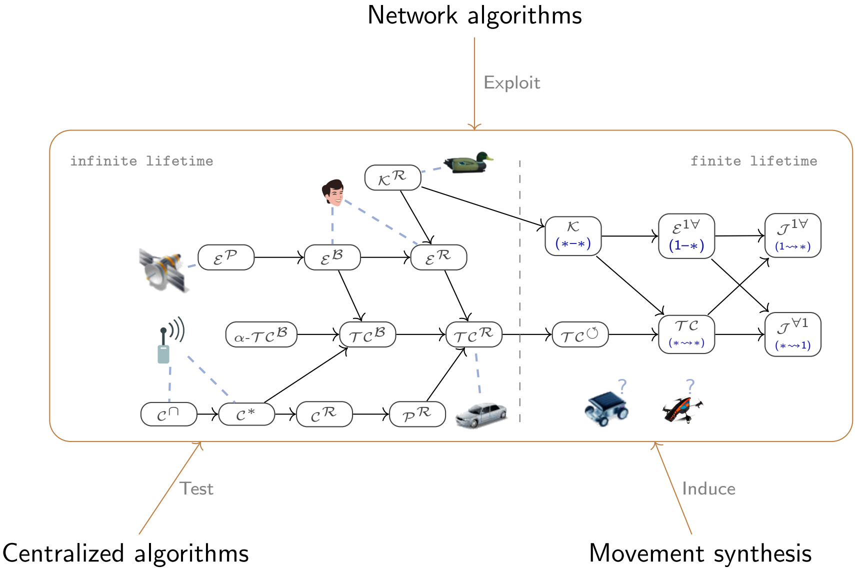
Robust MDS Non robust MDS





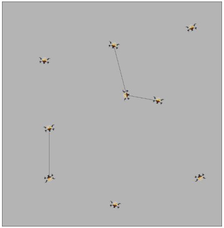
Local algorithm for ∀MIS

Classes of dynamic networks



Algorithmic movement synthesis

1) Collective movements which induce temporal structure



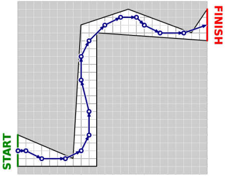
→ Synthesizing collective movements (*a.k.a.* *mobility models*) that satisfy temporal properties on the resulting communication graph (combined with a target mission).

← Ex: this network $\in \mathcal{E}^{\mathcal{R}}$

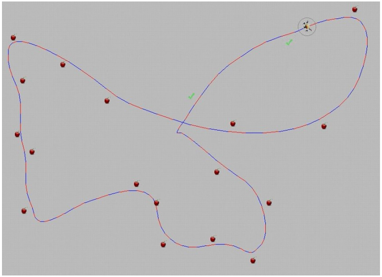
`(Credit video: Jason Schoeters)`

2) Integrating physical constraints in a tractable way

Discrete acceleration models
VECTOR RACER
(paper and pencil game)



→ Impact on problems, *e.g.* TSP ↗



Acceleration does impact the visit order!

Thank you!

The content of this talk is compiled in Chapters 2 and 3 of

A. Casteigts. *Finding Structure in Dynamic Networks*,

(Monograph available at `https://arxiv.org/abs/1807.07801`)