

# Impact of Network Dynamics on the Feasibility of Distributed Problems

## — Overview of Early Results

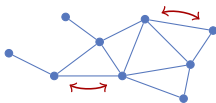
Arnaud Casteigts

—  
University of Bordeaux

21° Jornadas de Concurrencia y Sistemas Distribuidos (JCSD 2013)

UPV/EHU, San Sebastian

## Distributed Computing



Collaboration of distinct entities to perform a common task.

No centralization available. Direct interaction only.

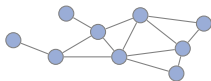
(Think globally, act locally)

# Examples of problems

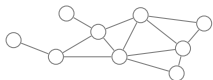
Broadcast



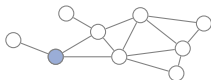
Propagating a piece of information from one node to all others.



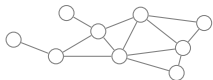
Election



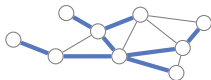
Distinguishing exactly one node among all.



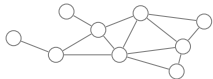
Spanning tree



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



Counting



Determining how many participants there are.



Consensus, naming, routing, exploration, ...

# Dynamic Networks



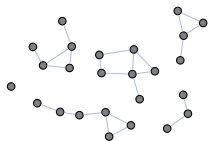
# Dynamic networks ?

In fact, *highly* dynamic networks.



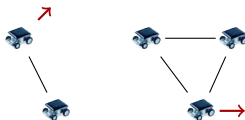
How changes are perceived ?

- Faults and Failures?
- Nature of the system. Change is normal.



Example of scenario

(say, exploration by mobile robots)



# Dynamic Graphs

Also called *evolving graphs* or *time-varying graphs*.

## Graph-centric

Sequence of static graphs  $\mathcal{G} = G_0, G_1, \dots$  [+table of association with dates in  $\mathbb{T}$ ]

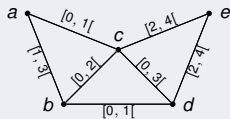


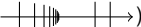
## Edge-centric

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

with  $\rho$  being a *presence function*

$(\rho : E \times \mathbb{T} \rightarrow \{0, 1\})$

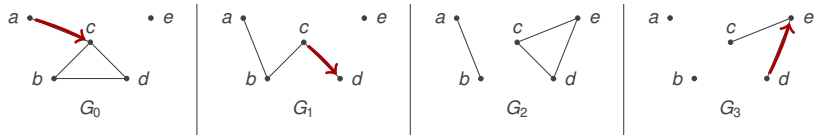


→ Both are theoretically equivalent if  $\rho$  is countable (e.g. not like this )

→ Further extensions possible (latency function, node-presence function, ...)

For references, see ([Ferreira, 2004](#)) and ([C., Flocchini, Quattrociocchi, Santoro, 2012](#))

# Basic graph concepts



⇒ Paths become temporal (a.k.a. *journey*)

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

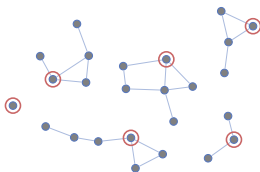
⇒ *Temporal connectivity*. Not symmetrical! (e.g.  $a \rightsquigarrow e$ , but  $e \not\rightsquigarrow a$ )

⇒ *Strict* journeys vs. *non-strict* journeys. (Important for analysis.)

In the literature : *Schedule-conforming path* (Berman, 1996); *Time-respecting path* (Kempe et al., 2008; Holme, 2005); *Temporal path* (Chaintriau et al., 2008); *Journey* (Bui-Xuan et al., 2003).

Many other concepts... (ask me!).

## Analytical approach



*Feasibility, Complexity, Correctness, Necessary conditions, ...*

General results & understandability.



# Abstracting Communications

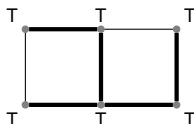
## Coarse-grain interaction

→ Atomic and localized

(Population protocols ([Angluin et al., 2004](#));

Graph relabeling systems ([Litovsky et al., 1999](#)))

Ex :  $T \text{ --- } N \longrightarrow T \text{ --- } T$  (*Spanning tree algorithm with a distinguished root*)



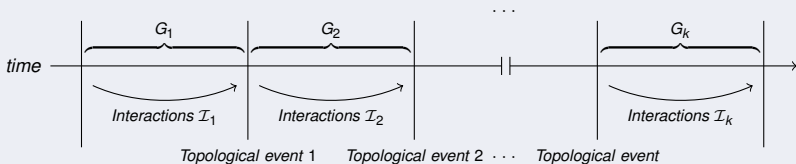
Note : In these models, scheduling is not part of the algorithm !

→ It is, e.g., probabilistic, adversarial, or even abstracted.

## Scope of the models

Relations between them ([Chalopin, 2006](#))



Interactions over a Dynamic Graph  $\mathcal{G} = \{G_1, G_2, \dots, G_k\}$ 

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

$$X = \mathcal{I}_k \circ \text{Event}_{k-1} \circ \dots \circ \text{Event}_2 \circ \mathcal{I}_2 \circ \text{Event}_1 \circ \mathcal{I}_1(G_0) \quad \text{Non deterministic !}$$

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

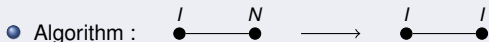
What makes a graph property  $\mathcal{P}$  a necessary or sufficient condition for success on  $\mathcal{G}$  ?

→ Necessary condition :  $\neg \mathcal{P}(\mathcal{G}) \implies \forall X \in \mathcal{X}, \text{failure}(X)$ .

→ Sufficient condition :  $\mathcal{P}(\mathcal{G}) \implies \forall X \in \mathcal{X}, \text{success}(X)$ . (+ schedul. assumption)

## Basic broadcast

- Initial states : I (source node), N (other nodes)
- Final states : I everywhere



Necessary condition for success ?

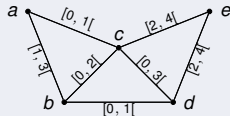
→  $\mathcal{P}_N$  : there exists a journey from the source node to all others (noted  $src \rightsquigarrow *$ ).

Sufficient condition ?

→  $\mathcal{P}_S$  : there exists a strict journey from the source to all others (noted  $src \rightsquigarrow^{st} *$ ).

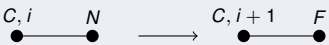
## Classes of dynamic graphs

- $\mathcal{C}_1$  :  $\mathcal{P}_N$  is satisfied by at least one node (noted  $1 \rightsquigarrow *$ ).
- $\mathcal{C}_2$  :  $\mathcal{P}_N$  is satisfied by all nodes ( $* \rightsquigarrow *$ ).
- $\mathcal{C}_3$  :  $\mathcal{P}_S$  is satisfied by at least one node ( $1 \rightsquigarrow^{st} *$ ).
- $\mathcal{C}_4$  :  $\mathcal{P}_S$  is satisfied by all nodes ( $* \rightsquigarrow^{st} *$ ).



## Counting with a selected counter

- Initial states :  $(C, 1)$  (counter node),  $N$  (other nodes).
- Final states :  $(C, |V|)$  (counter node).

• Algorithm : 

## Necessary or sufficient conditions

- $\mathcal{P}_N$  : there exists an edge, at some time, between the counter and every other node (noted  $cpt \rightarrow *$ ).
- $\mathcal{P}_S = \mathcal{P}_N$ .

## Classes of dynamic graphs

- $\mathcal{C}_5$  : at least one node verifies  $\mathcal{P}$ , (noted  $1 \rightarrow *$ ).
- $\mathcal{C}_6$  : all the nodes verify  $\mathcal{P}$ , (noted  $* \rightarrow *$ ).

## Tight Necessary condition

- Not satisfied  $\implies$  failure is guaranteed  $(\nexists X, success(X))$
- *Satisfied*  $\implies$  *success is possible*  $(\exists X, success(X)).$

## Tight Sufficient condition

- Satisfied  $\implies$  success is guaranteed  $(\nexists X, failure(X))$
- *Not satisfied*  $\implies$  *failure is possible*  $(\exists X, failure(X))$

Remark : necessary and sufficient conditions do not always exist !

Ex. basic broadcast :

- $\rightarrow src \rightsquigarrow *$  is a tight necessary condition (i.e. maximal)
- $\rightarrow src \overset{st}{\rightsquigarrow} *$  is a tight sufficient condition (i.e. minimal)

In between : outcome is uncertain... might succeed or fail (depending on scheduling/adversary).

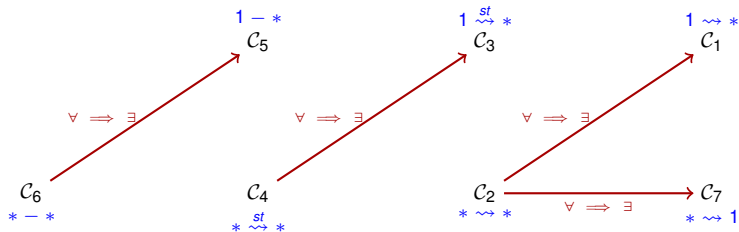
## Counting without a selected counter

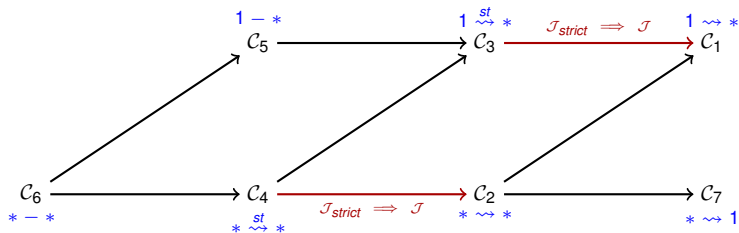
(Angluin et al., 2004)

- Initial states : 1 (all nodes).
- Final states :  $|V|$  (one node), 0 (anywhere else).
- Algorithm :  $\begin{array}{ccc} i \neq 0 & j \neq 0 & \\ \bullet & \text{---} & \bullet \end{array} \longrightarrow \begin{array}{ccc} i + j & & 0 \\ \bullet & \text{---} & \bullet \end{array}$

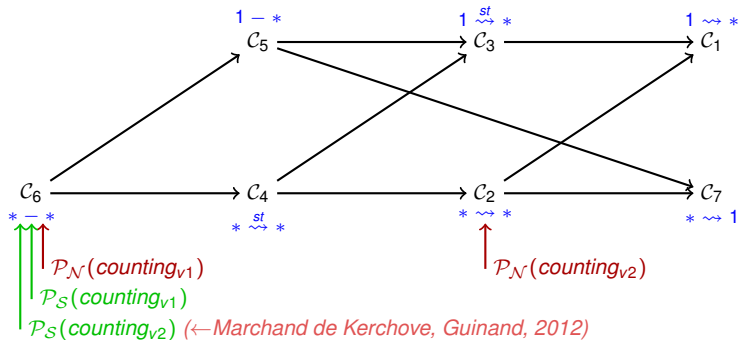
## Conditions and classes of graphs

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







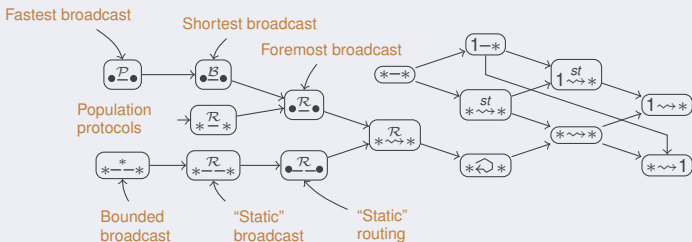


- Comparison of algorithms on a formal basis
- Decision making (what algorithm to use ?)
  - e.g. using automated property checking on network traces).
- Formal proofs ? (Coq)

Q : How far beyond toy examples ?



## Extending the hierarchy



Ex : Bounded broadcast in  $(*-*)$

(O'Dell and Wattenhofer, 2005)

The graph is arbitrarily dynamic, as long as every  $G_i$  remains connected :

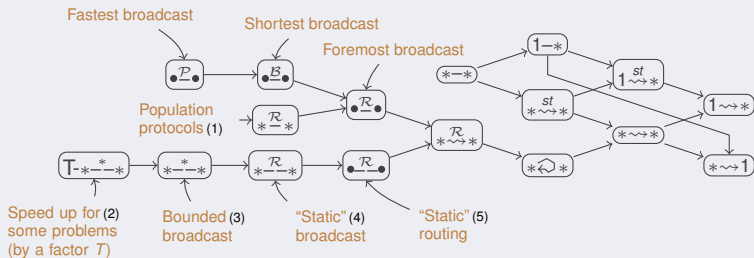


Min cut of size 1 between informed and uninformed nodes :

→ At least one new node informed in each step.



## Extending the hierarchy










- (1) Complete graph of interaction (Angluin, Aspnes, Diamadi, Fischer, Peralta, 2004)
- (2) T-interval connectivity (Kuhn, Lynch, Oshman, 2010)
- (3) Constant connectivity (O'Dell and Wattenhofer, 2005)
- (4) Eventual connectivity (Ramanathan, Basu, and Krishnan, 2007)
- (5) Eventual routability (Ramanathan, Basu, and Krishnan, 2007)



# Gracias, Milesker, Merci !

## References (external) :

-  I. Litovsky, Y. Métivier, and E. Sopena., [Graph relabelling systems and distributed algorithms.](#), *Handbook of graph grammars and computing by graph transformation*, 1999.
-  F. Marchand de Kerchove and F. Guinand., [Strengthening Topological Conditions for Relabeling Algorithms in Evolving Graphs](#), Technical report, Université Le Havre, 2012.
-  B. Bui-Xuan, A. Ferreira, and A. Jarry., [Computing shortest, fastest, and foremost journeys in dynamic networks](#), *JFCS* 14(2) : 267-285, 2003.
-  A. Ferreira, [Building a Reference Combinatorial Model for manets](#), *IEEE Network* 18(5) : 24-29, 2004.
-  R. O'Dell and R. Wattenhofer, [Information dissemination in highly dynamic graphs](#), *DIALM-POMC*, 2005.
-  F. Kuhn, N. Lynch, R. Oshman, [Distributed computation in dynamic networks](#), *STOC*, 2010.
-  R. Ramanathan, P. Basu, and R. Krishnan, [Towards a Formalism for Routing in Challenged Networks](#), *CHANTS*, 2007.
-  D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, R. Peralta, [Computation in networks of passively mobile finite-state sensors](#), *PODC*, 2004.
-  J. Chalopin, [Algorithmique Distribuée, Calculs Locaux et Homomorphismes de Graphes](#), PhD Thesis, University of Bordeaux, 2006.

## References (internal) :

-  A. Casteigts, S. Chaumette, A. Ferreira., [Characterizing Topological Assumptions of Distributed Algorithms in Dynamic Networks](#), *SIROCCO*, 2009. (Long version in *CoRR* abs/1102.5529, 2012.)
-  A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro., [Time-Varying Graphs and Dynamic Networks.](#), *IJPEDES* 27(5) : 387-408, 2012.
-  A. Casteigts, P. Flocchini, B. Mans, N. Santoro., [Measuring Temporal Lags in Delay-Tolerant Networks](#), *IEEE Transactions on Computer*, 2013.
-  A. Casteigts, P. Flocchini, B. Mans, N. Santoro., [Deterministic computations in time-varying graphs : Broadcasting under unstructured mobility](#), *IFIP TCS*, 2010. (Long version in *CoRR* abs/1210.3277, 2012.)