DYNAMIC NETWORKS

(Dynamic of networks)

DYNAMIC NETWORKS

- Most real world networks are dynamic
  - Facebook friendship
    - People joining/leaving
    - Friend/Unfriend
  - Twitter mention network
    - Each mention has a timestamp
    - Aggregated every day/month/year => still dynamic
  - World Wide Web
  - Urban network
  - …

DYNAMIC NETWORKS

- Most real world networks are dynamic
  - Nodes can appear/disappear
  - Edges can appear/disappear
  - Nature of relations can change
- How to represent those changes?
- How to manipulate dynamic networks?

DYNAMIC NETWORKS

Semantic level

Relations

Long term

-Friend
-Colleague
-Family relation
-…

Short term ?

-Collaborators in the same project
-Same team in a game
-Attendees of the same meeting
-…

Interactions

Instantaneous

-e-mail
-Text message
-Co-authoring
…

With duration

-Phone call
-Discussion in real life
-Participate in a same meeting

DYNAMIC NETWORKS

Semantic level

Relations

Interactions

Representation level

Interval graphs

DN=(V,E,T,DV)
DV:VxTxT
E:VxVxTxT

Graph series

DN=[G1,G2…Gn]
Gi=(V,E)
E :VxV

(Or 3D tensor)

Link Streams

DN=(V,E,T)
E:VxVxT

DYNAMIC NETWORKS

Semantic level

Representation level

Relations

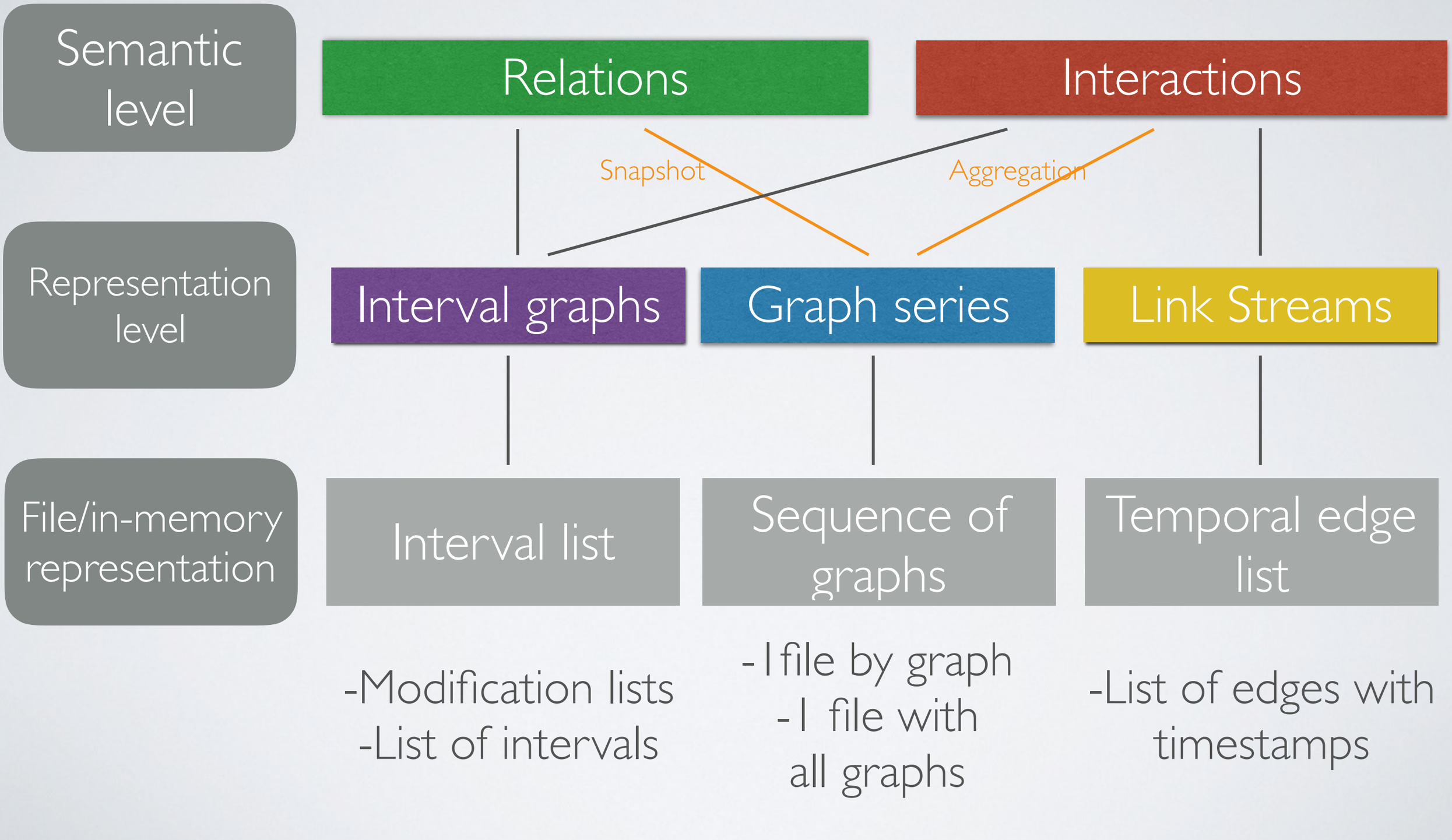
Interactions

Snapshot

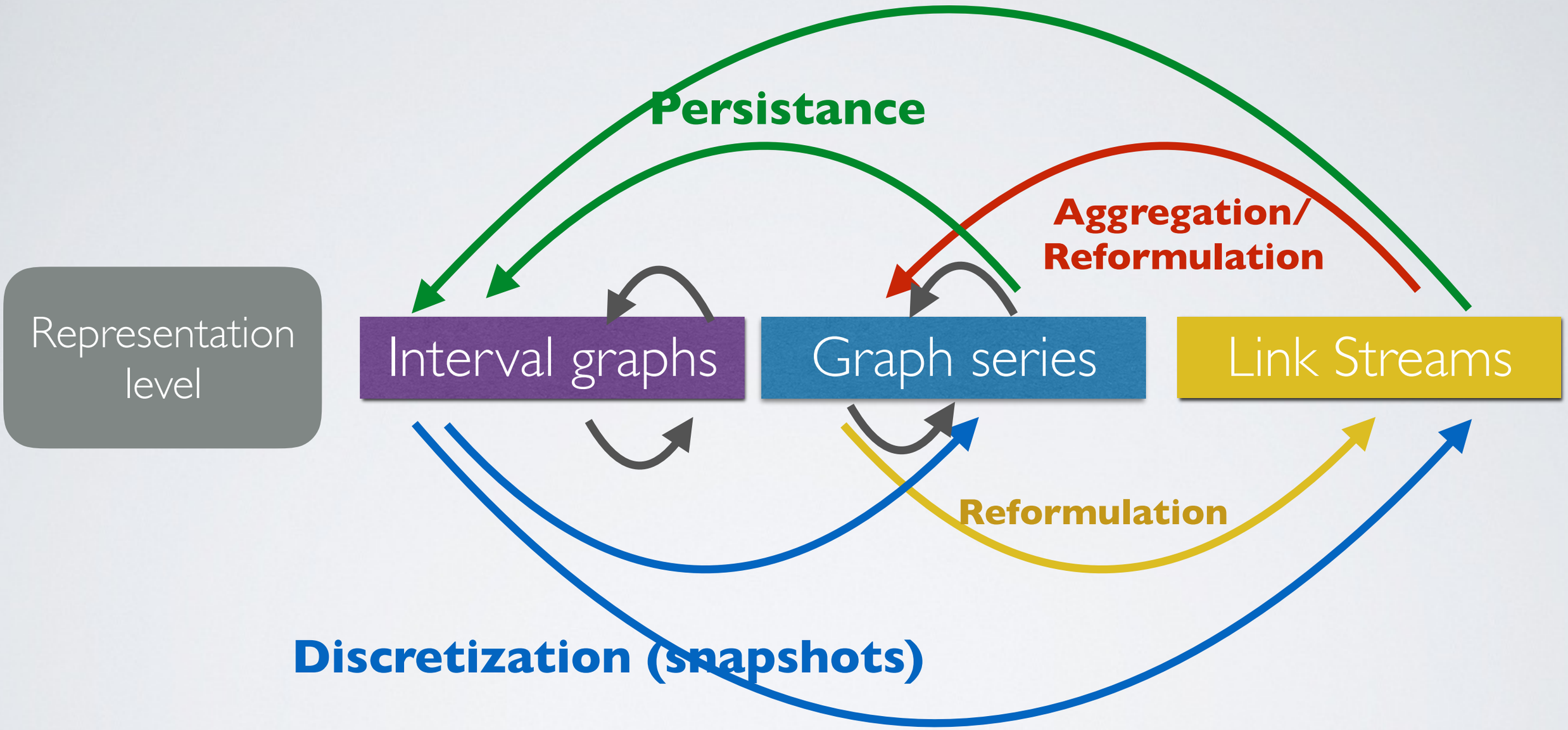
Aggregation

Interval graphs	Graph series	Link Streams
DN=(V,E,T,DV) DV:VxTxT E:VxVxTxT	DN={G1,G2…Gn} Gi=(V,E) E :VxV	DN=(V,E,T) E:VxVxT

DYNAMIC NETWORKS



DYNAMIC NETWORKS



DYNAMIC NETWORKS

- Exemple in practice: Sociopattern dataset
  - Every 20s, list of individuals at distance ≈ 1,5m
  - Dataset : **sequence of graphs** or **temporal edge list**

```
1353304100      1148 1644
1353304100      1613 1672
1353304100      656  682
1353304100      1632 1671

1353304120      1492 1613
1353304120      656  682
1353304120      1632 1671

1353304140      1148 1644

1353304160      656  682
1353304160      1108 1601
1353304160      1632 1671
1353304160      626  698
```

Types of network evolution

According to

1) Observation frequency
2) Network nature

Relations

The graph is more and more stable, until most observations are completely similar to previous/later ones (frequency faster than change rate)

Static Network → **Higher Observation Frequency**

The graph is less and less stable, until each observation is a graph in itself, thus completely different from previous/later ones (frequency faster than observed events rate)

Exhaustive / continuous time

Interactions

ANALYZING DYNAMIC NETWORKS

DISTINGUISHING:

-UNSTABLE SNAPSHOTS

-STABLE NETWORKS

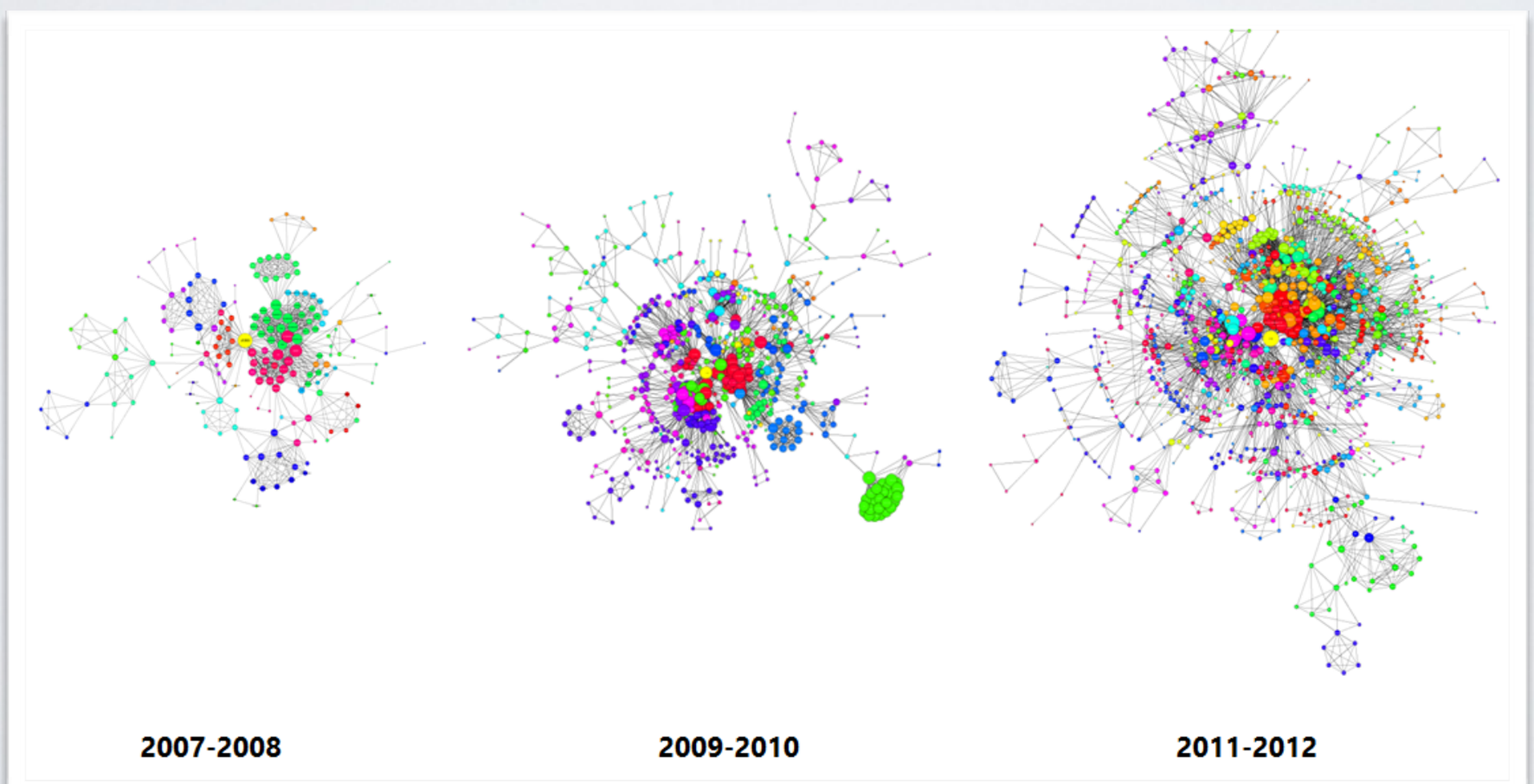
-(UNSTABLE) TEMPORAL NETWORKS

(WITH OR WITHOUT DURATION)

UNSTABLE SNAPSHOTS

UNSTABLE SNAPSHOTS

- The evolution is represented as a series of *a few* snapshots.
- Many changes between snapshots
  - Cannot be visualized as a "movie"



UNSTABLE SNAPSHOTS

- Each snapshot can be studied as a static graph
- The evolution of the properties can be studied "manually"
- "Node X had low centrality in snapshot t and high centrality in snapshot t+n"

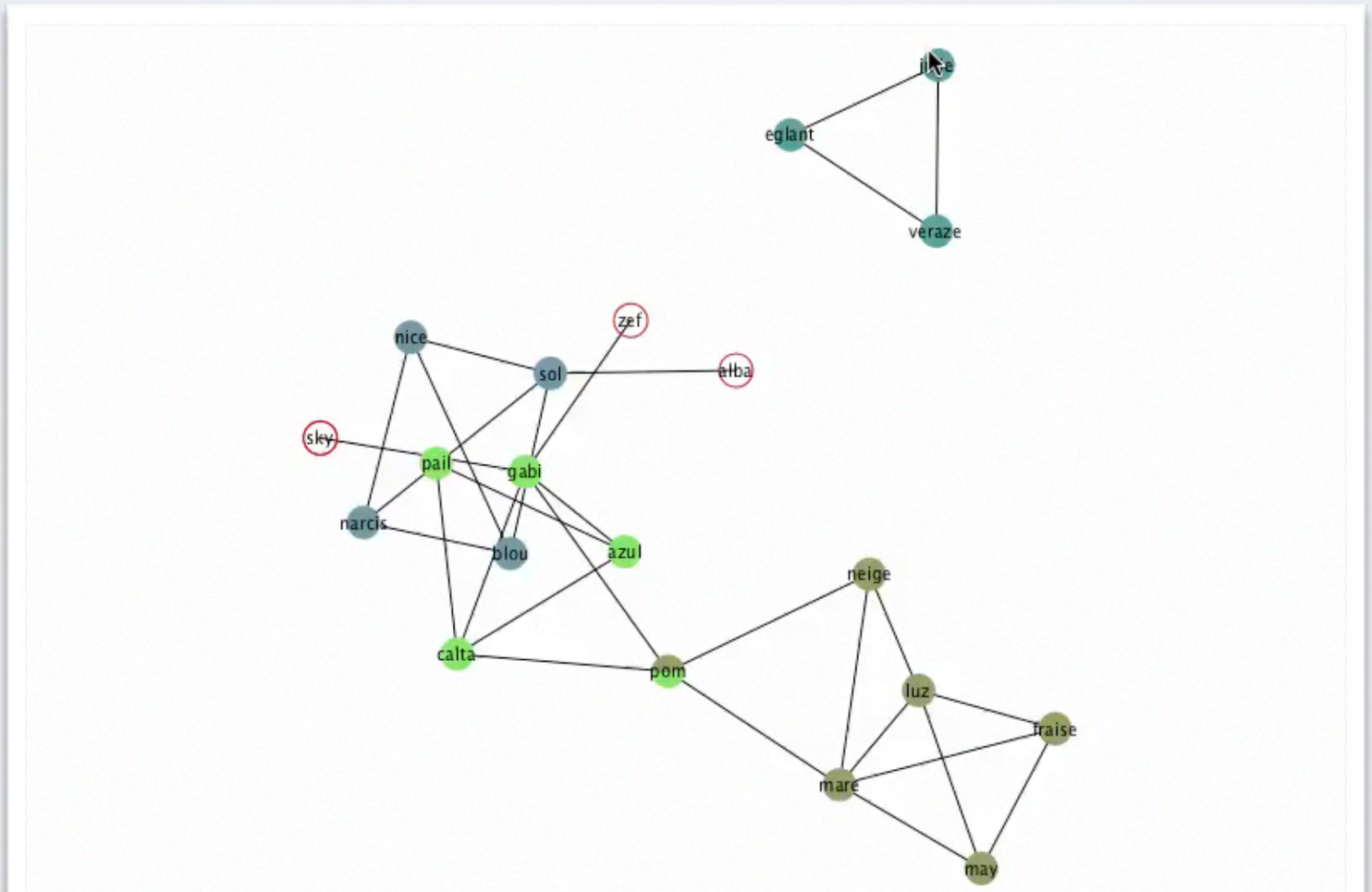
STABLE NETWORKS

STABLE NETWORK

- Edges change (relatively) slowly
- The network is well defined at any t
  - Temporal network: nodes/edges described by (long lasting) intervals
  - Enough snapshots to track nodes
- A static analysis at every (relevant) t gives a dynamic vision
- No formal distinction with previous case (higher observation frequency)

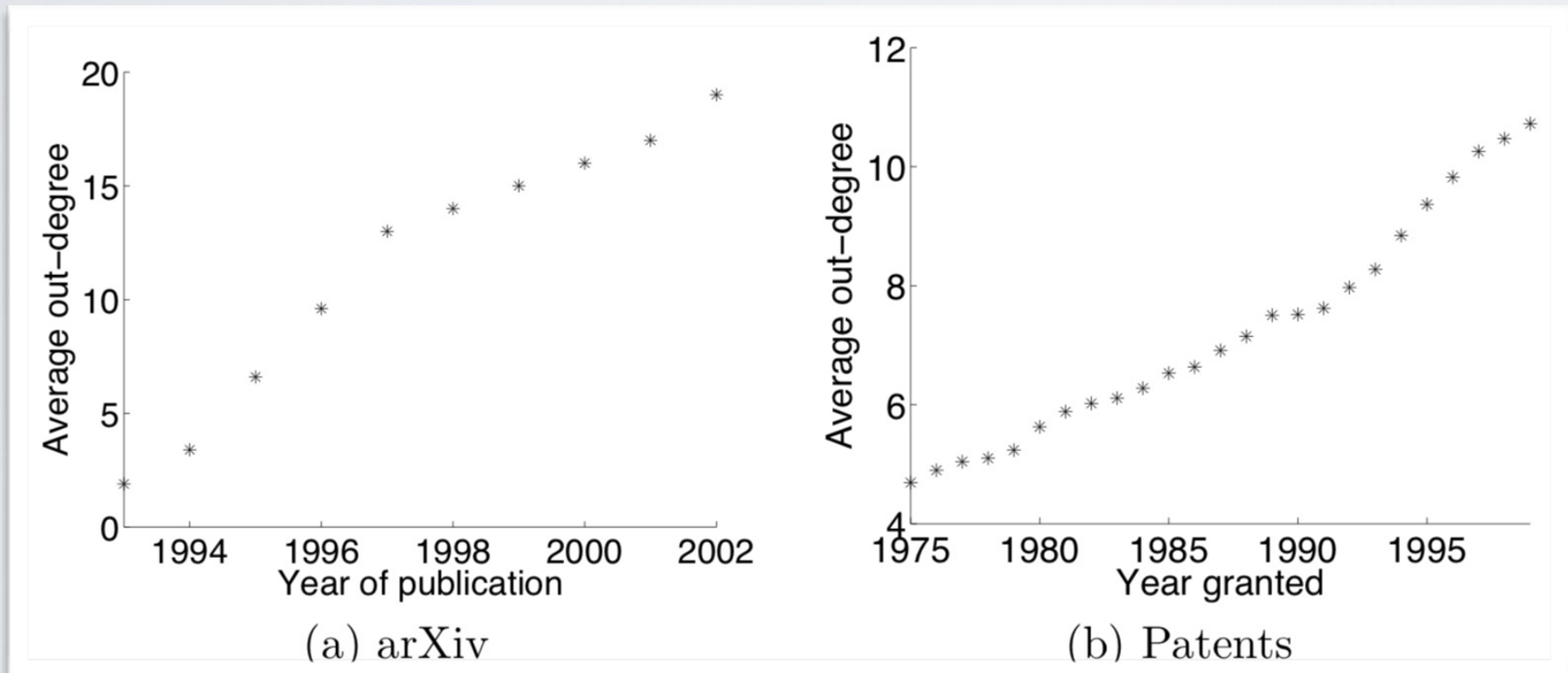
STABLE NETWORK

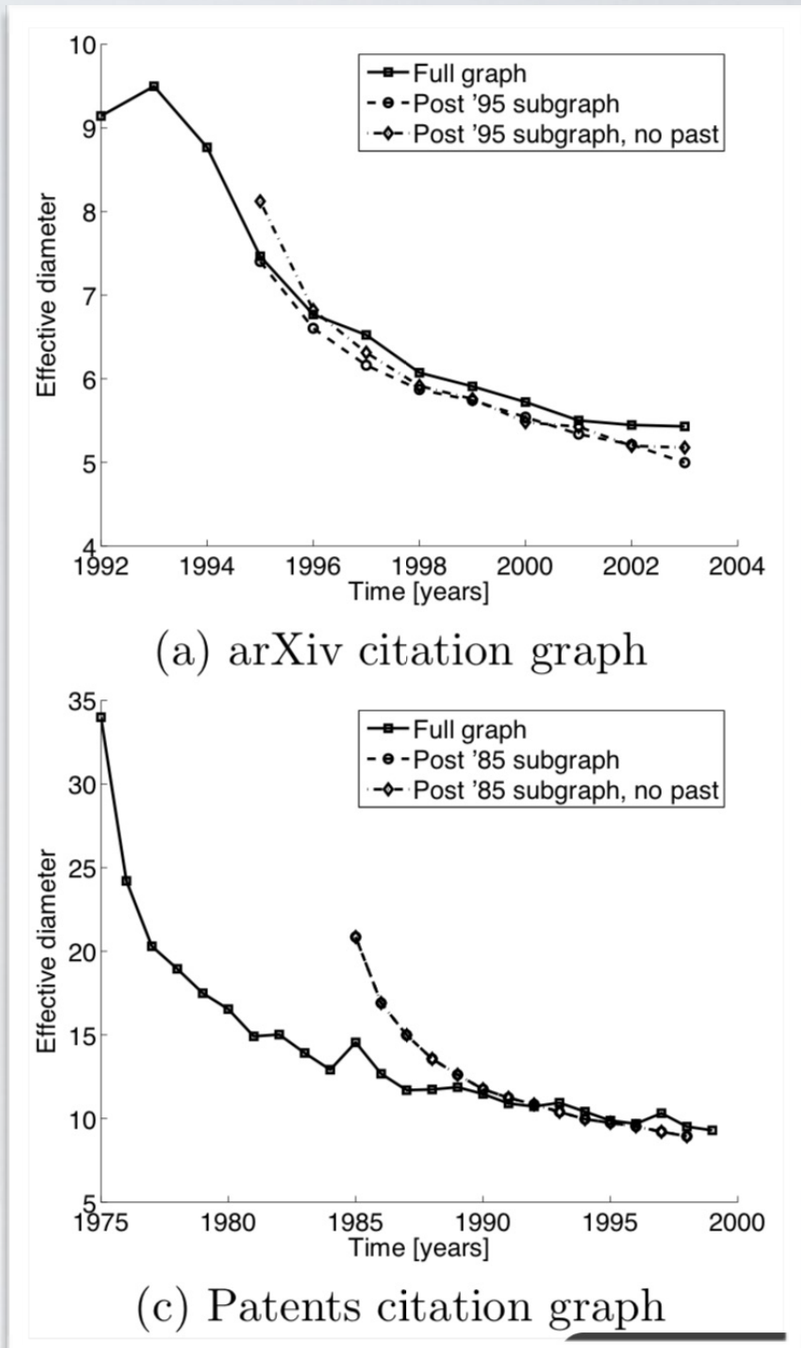
- Visualization
  - Problem of stability of node positions



STABLE NETWORK

- Global graph properties

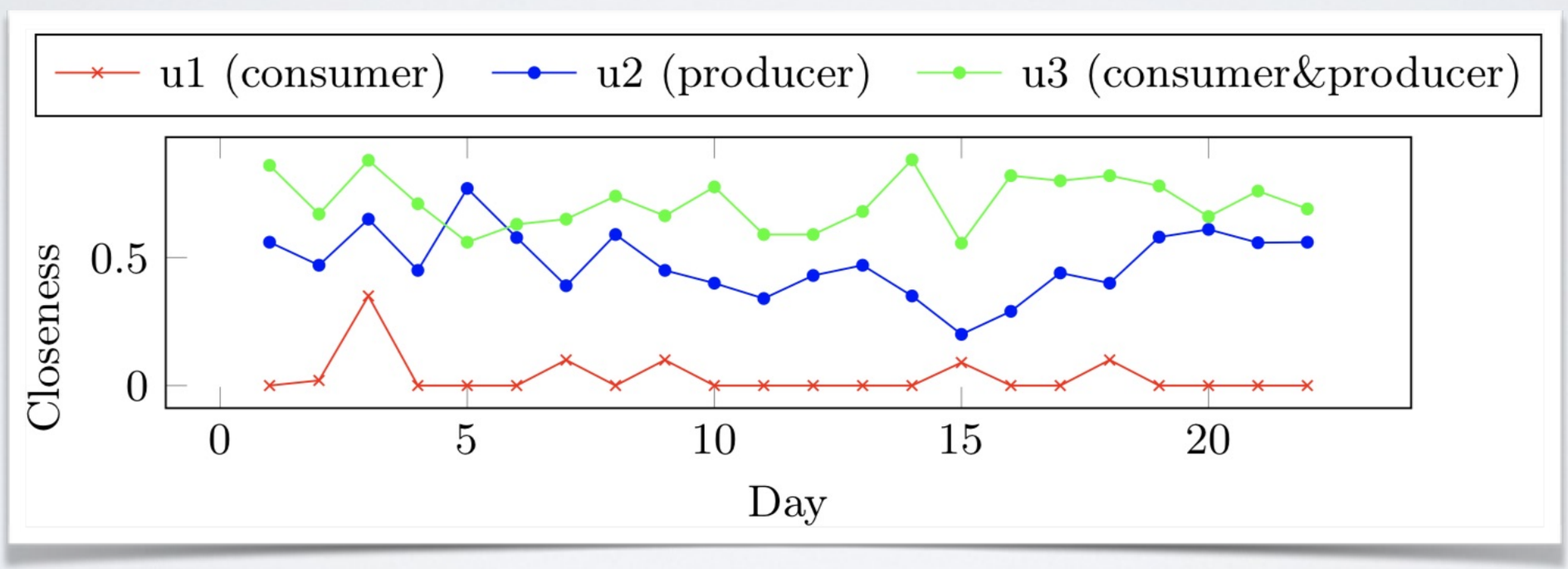




Leskovec, Jure, Jon Kleinberg, and Christos Faloutsos. "Graph evolution: Densification and shrinking diameters." *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007): 2.

STABLE NETWORK

- Centralities



TIME SERIES ANALYSIS

- TS analysis is a large field of research
- Time series: evolution of a value over time
  - Stock market, temperatures…
- "Killer app":
  - Detection of periodic patterns
  - Detection of anomalies
  - Identification of global trends
  - Evaluation of auto-correlation
  - Prediction of future values
- e.g. ARIMA (Autoregressive integrated moving average)

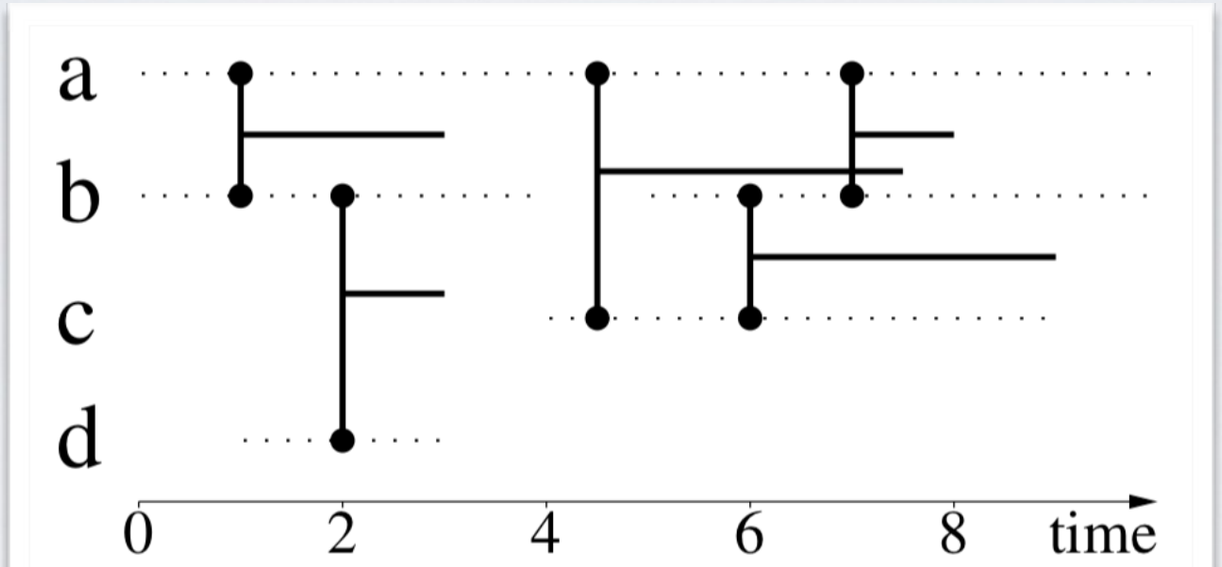
https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average

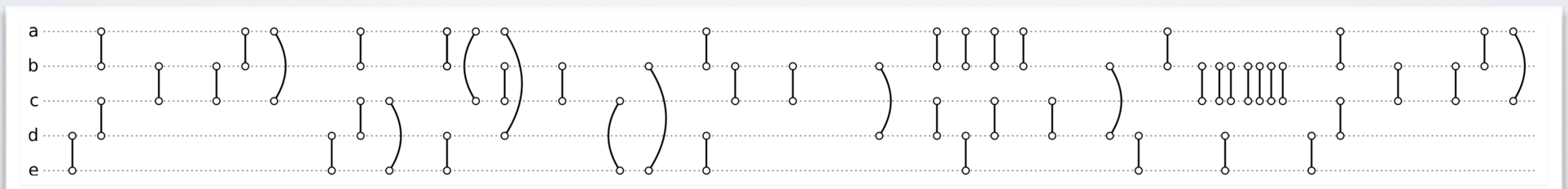
UNSTABLE TEMPORAL NETWORKS

UNSTABLE TEMPORAL NETWORK

- The network at a given *t* is not meaningful
- How to analyze such a network?

UNSTABLE TEMPORAL NETWORK





UNSTABLE TEMPORAL NETWORK

- Until recently, network was transformed using aggregation/ sliding windows
  - Information loss
  - How to chose a proper aggregation window size?

- Tools developed to deal with such networks

UNSTABLE TEMPORAL NETWORK

- [Holme 2012]: mostly about paths, walks, distances… (later class, diffusion on networks.)

Holme, Petter, and Jari Saramäki. "Temporal networks." *Physics reports* 519.3 (2012): 97-125.

- [Latapy 2018]: Other things (centralities, …)

Latapy, M., Viard, T., & Magnien, C. (2018). Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, *8*(1), 61.

- Idea: Generalize all graphs definitions to temporal networks

- => If all nodes and all edges always present, same values as for a static graph

CENTRALITIES & NETWORK PROPERTIES IN STREAM GRAPHS

STREAM GRAPHS

$$\textbf{stream graph } S = (T, V, W, E)$$

T: Possible Time

V: vertices

W: Vertices presence time

E: Edges presence time

INDICES IN STREAM GRAPHS

Number of nodes:

Total presence of nodes

Total dataset duration

(not an integer value…)

$$n = \sum_{v \in V} n_v = \frac{|W|}{|T|}$$

e.g.: 2 if 4 nodes
half the time

INDICES IN STREAM GRAPHS

Number of edges:

Total presence of edges

Total dataset duration

(not an integer value…)

$$m = \sum_{uv \in V \otimes V} m_{uv} = \frac{|E|}{|T|}$$

e.g.: 1 if 1 edge all the time

INDICES IN STREAM GRAPHS

Neighborhood of a node

$$N(v) = \{(t, u), (t, uv) \in E\}$$

Degree of a node

$$d(v) = \frac{|N(v)|}{|T|} = \sum_{u \in V} \frac{|T_{uv}|}{|T|}$$

INDICES IN STREAM GRAPHS

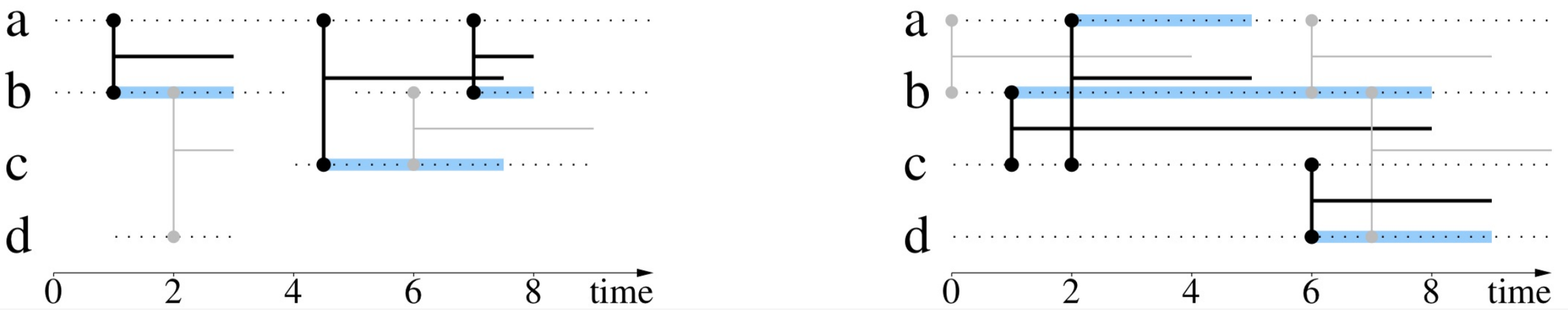


Figure 5: **Two examples of neighborhoods and degrees of nodes.** We display in black the links involving the node under concern, and in grey the other links. Left: $N(a) = ([1,3] \cup [7,8]) \times \{b\} \cup [4.5,7.5] \times \{c\}$ is in blue, leading to $d(a) = \frac{3}{10} + \frac{3}{10} = 0.6$. Right: $N(c) = [2,5] \times \{a\} \cup [1,8] \times \{b\} \cup [6,9] \times \{d\}$ is in blue, leading to $d(c) = \frac{13}{10} = 1.3$.

INDICES IN STREAM GRAPHS

Average node degree

$$d(V) = \frac{1}{n} \cdot \sum_{v \in V} n_v \cdot d(v) = \sum_{v \in V} \frac{|T_v|}{|W|} \cdot d(v)$$

INDICES IN STREAM GRAPHS

Clustering coefficient of a node

$$cc(v) = \delta(N(v)) = \frac{\sum_{uw \in V \otimes V} |T_{vu} \cap T_{vw} \cap T_{uw}|}{\sum_{uw \in V \otimes V} |T_{vu} \cap T_{vw}|}$$

Probability that if we take 2 random neighbors at a random time, they are linked

INDICES IN STREAM GRAPHS

$$\delta(S) = \frac{\sum\limits_{uv \in V \otimes V} |T_{uv}|}{\sum\limits_{uv \in V \otimes V} |T_u \cap T_v|} = \frac{\int\limits_{t \in T} |E_t| \, \mathrm{d}t}{\int\limits_{t \in T} |V_t \otimes V_t| \, \mathrm{d}t}$$

Density (of a stream graph): probability if we take a random pair of nodes at a random time that there is an edge between them

INDICES IN STREAM GRAPHS

$$\delta(S) = \frac{\sum_{uv \in V \otimes V} |T_{uv}|}{\sum_{uv \in V \otimes V} |T_u \cap T_v|} = \frac{\int_{t \in T} |E_t| \, \mathrm{d}t}{\int_{t \in T} |V_t \otimes V_t| \, \mathrm{d}t}$$

Total edge presence

e.g.: 10 if 2 edges present over 5 periods

INDICES IN STREAM GRAPHS

$$\delta(S) = \frac{\sum_{uv \in V \otimes V} |T_{uv}|}{\sum_{uv \in V \otimes V} |T_u \cap T_v|} = \frac{\int_{t \in T} |E_t| \, \mathrm{d}t}{\int_{t \in T} |V_t \otimes V_t| \, \mathrm{d}t}$$

Total **overlapping time** between each pair of nodes

=>An edge is possible

INDICES IN STREAM GRAPHS

Figure 2: **Two stream graphs with $n = 2$ nodes, $m = 1$ link, but with different densities:** Left: $\delta = 0.75$. Right: $\delta = 1$.

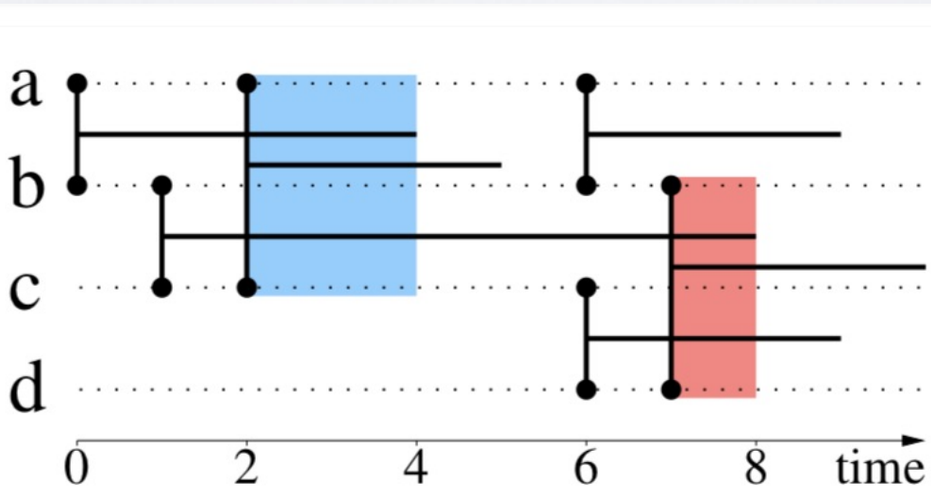
INDICES IN STREAM GRAPHS

- Note that we can define particular cases of density:
  - Density for a pair of nodes
  - Density for a node

$$\delta(uv) = \frac{|T_{uv}|}{|T_u \cap T_v|}, \quad \delta(v) = \frac{\sum_{u \in V, u \neq v} |T_{uv}|}{\sum_{u \in V, u \neq v} |T_u \cap T_v|}$$

INDICES IN STREAM GRAPHS

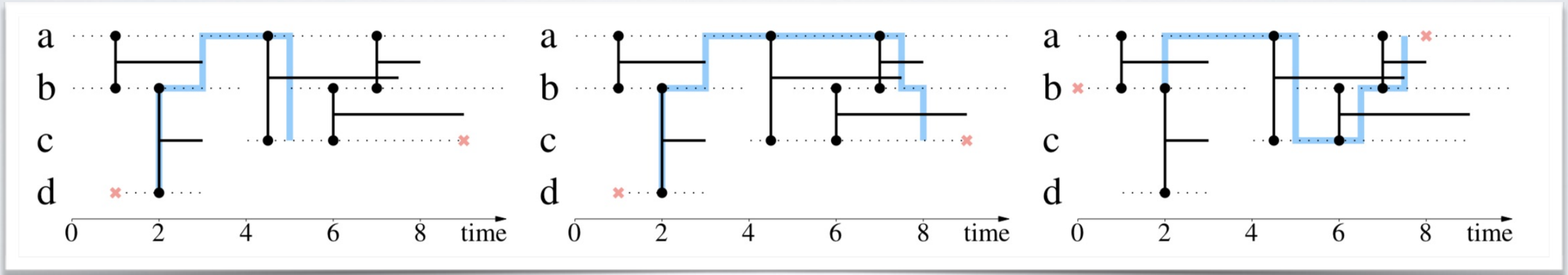
A clique of graph G is a cluster C of G of density 1. *In other words, all pairs of nodes involved in C are linked together in G. A clique C is maximal if there is no other clique C' such that $C \subset C'$.*



PATHS AND DISTANCES IN STREAM GRAPHS

PATHS

- A path in a stream graphs
  - Starts at a node and a date
  - Ends at a node and a date
  - Has a length (number of hops)
  - Has a duration (duration from leaving node to reaching node)



Path(d,1)(c9)
Length:3
Duration: 3

?

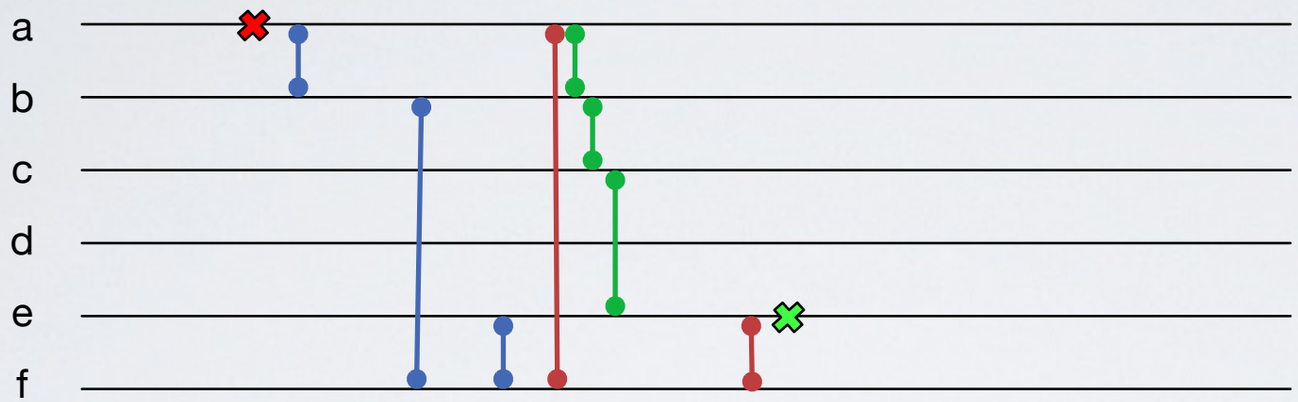
?

SHORTEST PATHS

- Several types of shortest paths in Stream graphs:
  - Shortest path: minimal length
  - Fastest path: minimal duration
  - Foremost path: first to reach

  - Fastest shortest paths
    - Minimum duration among minimal length
  - Shortest fastest paths
    - Minimal length among minimal duration

SHORTEST PATHS

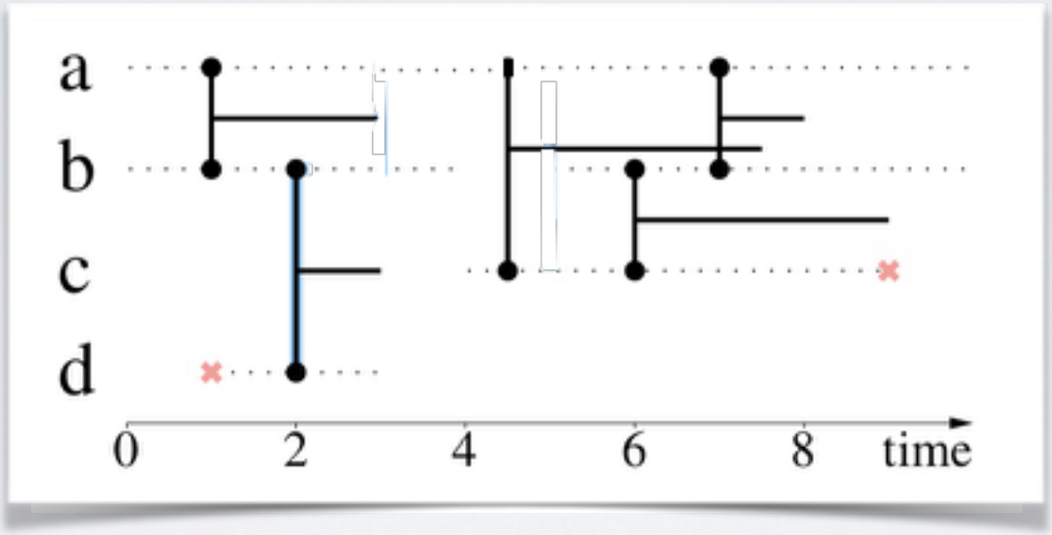


Blue: Foremost

Green: Fastest

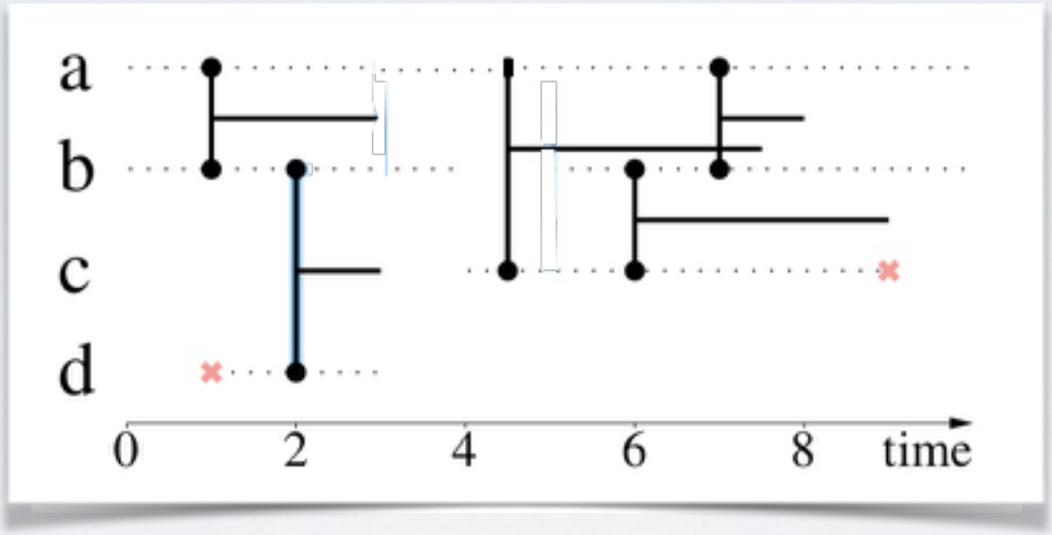
Red: Shortest

SHORTEST PATHS



Shortest paths from (1, d) to (9, c) ?

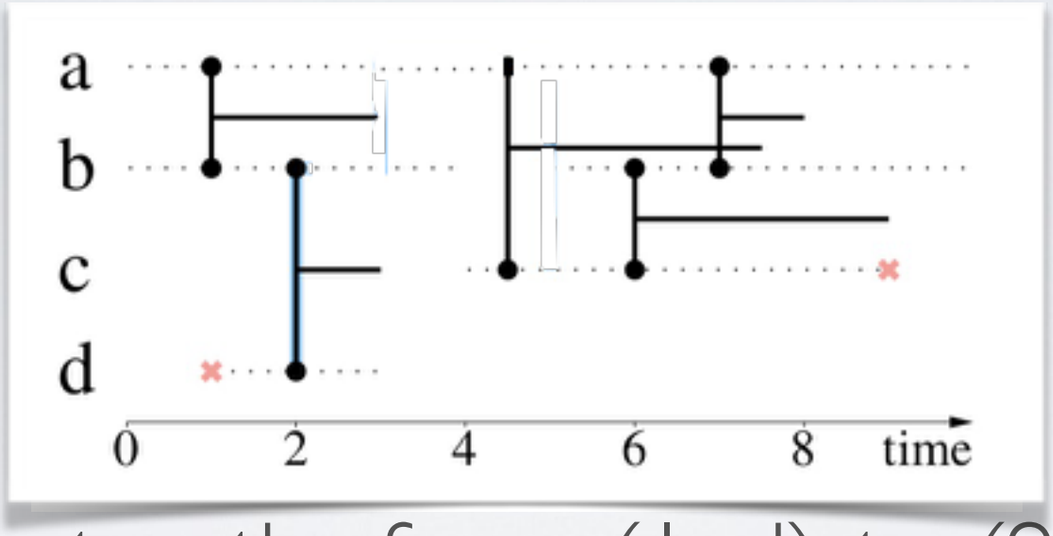
SHORTEST PATHS



Shortest paths from (1, d) to (9, c) ?

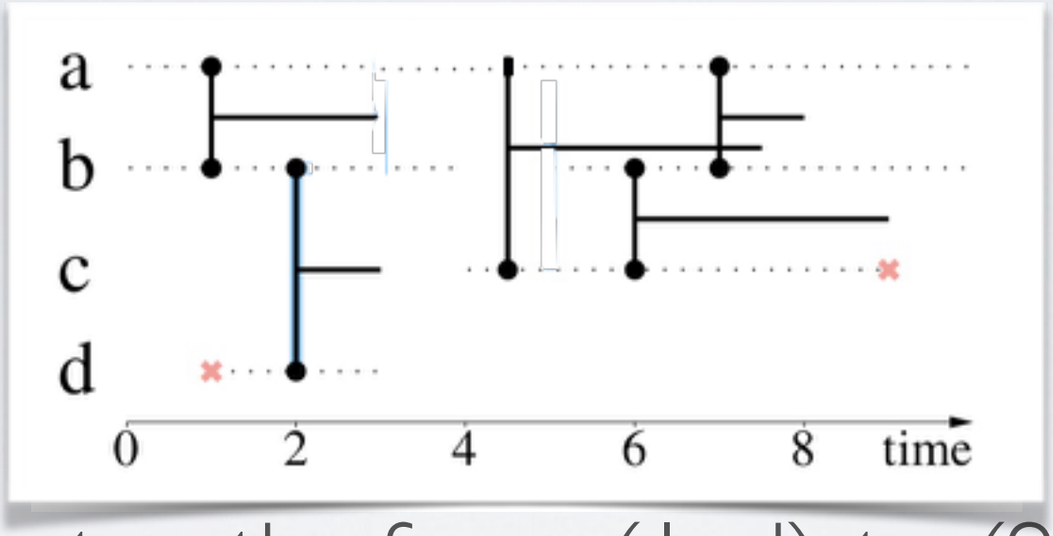
=>e.g. (2.5,d,b)(3,b,a)(7,a,c)

SHORTEST PATHS



Fastest paths from (1, d) to (9, c) ?

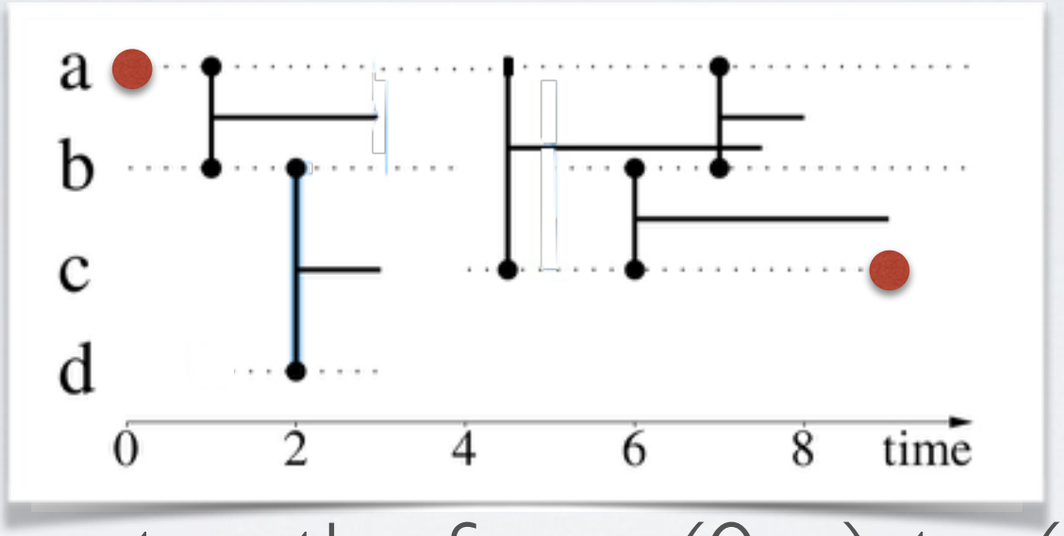
SHORTEST PATHS



Fastest paths from (1, d) to (9, c) ?

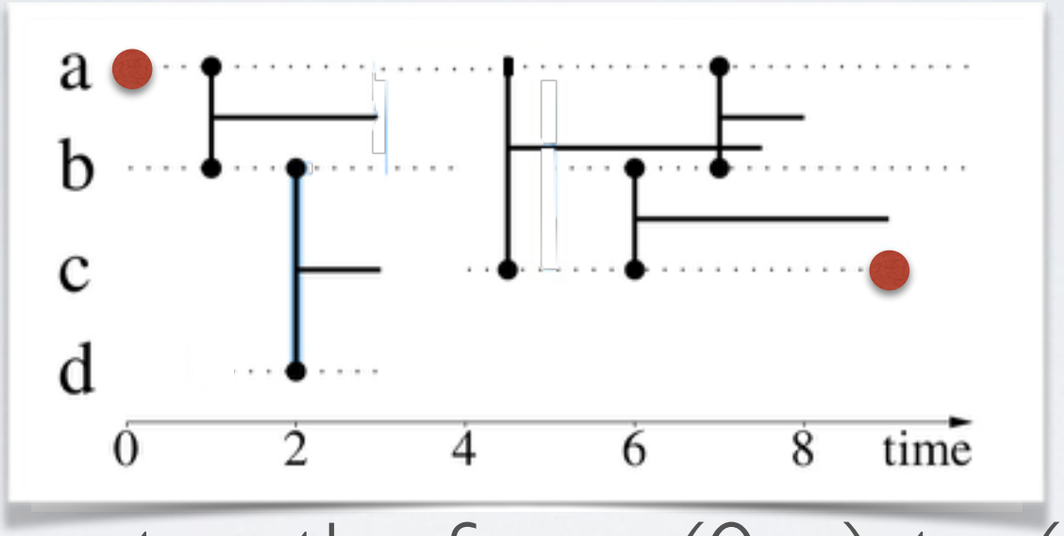
(3,d,b),(3,b,a),(4.5,a,c)

SHORTEST PATHS



Foremost paths from (0, a) to (9, c) ?

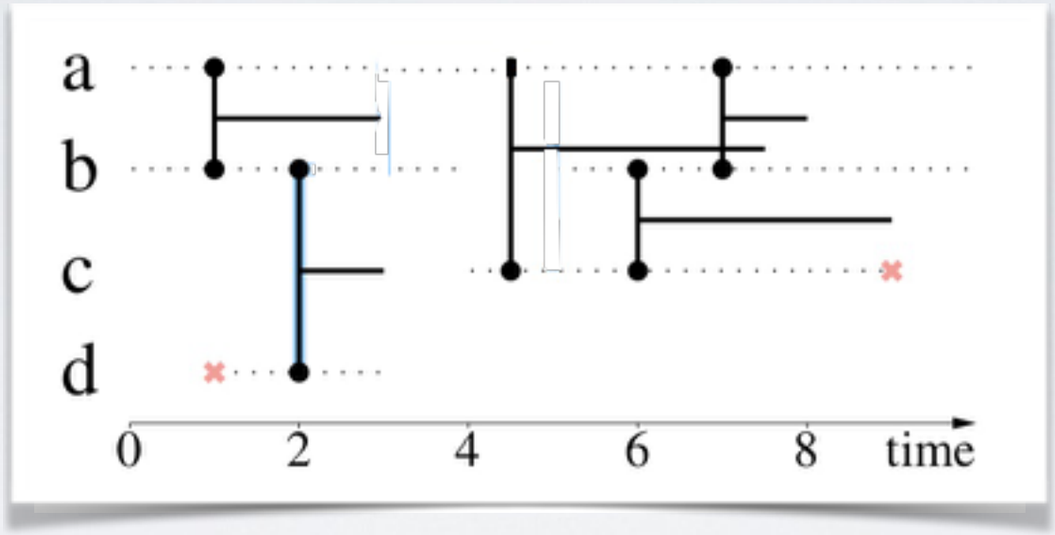
SHORTEST PATHS



Foremost paths from (0, a) to (9, c) ?

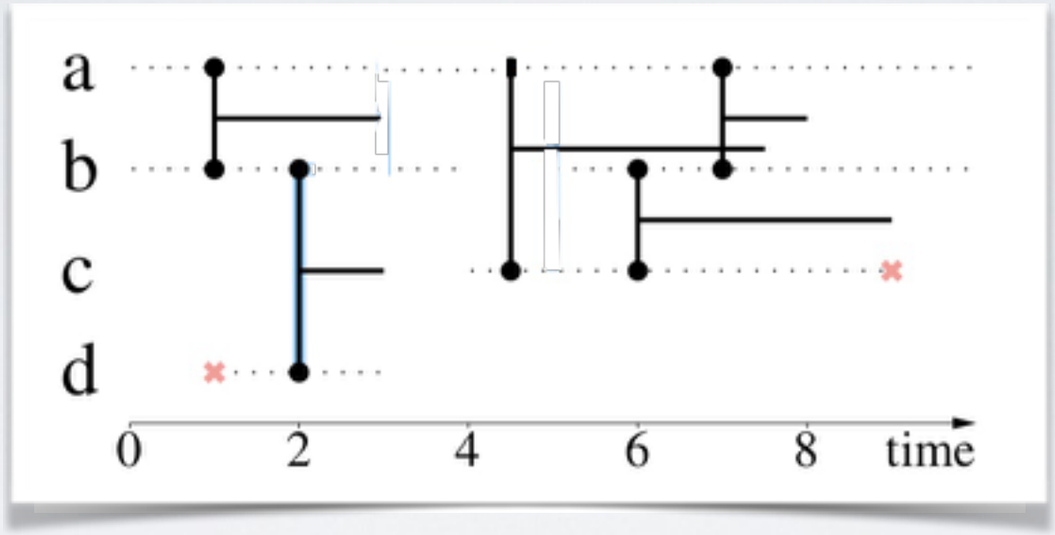
…(4.5,a,c)

SHORTEST PATHS



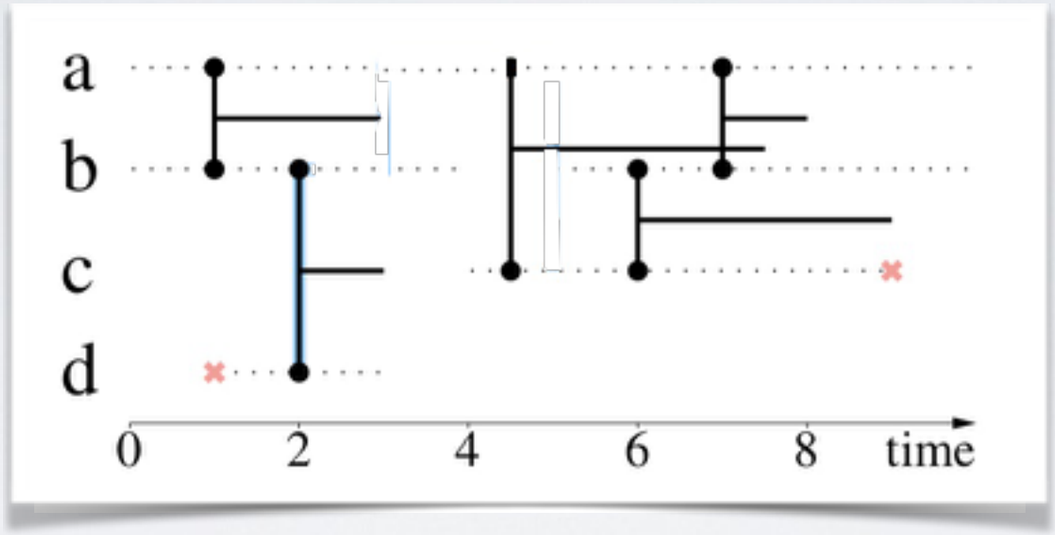
Fastest shortest path from (1, d) to (9, c) ?

SHORTEST PATHS



Fastest shortest path from (1, d) to (9, c) ?

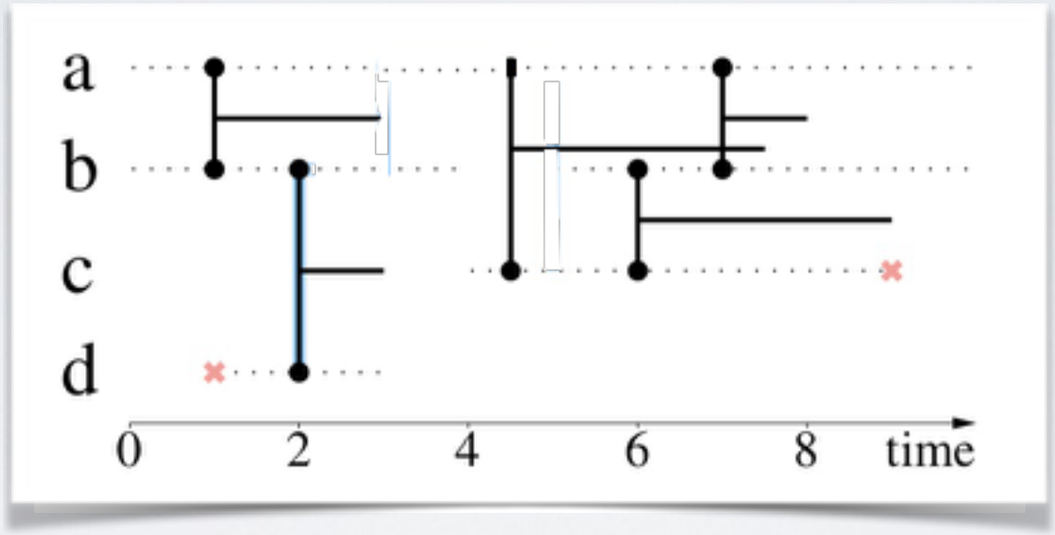
SHORTEST PATHS



Fastest Shortest path from (1, d) to (9, c) ?

(3, d, b), (3, b, a), (4.5, a, c)

SHORTEST PATHS

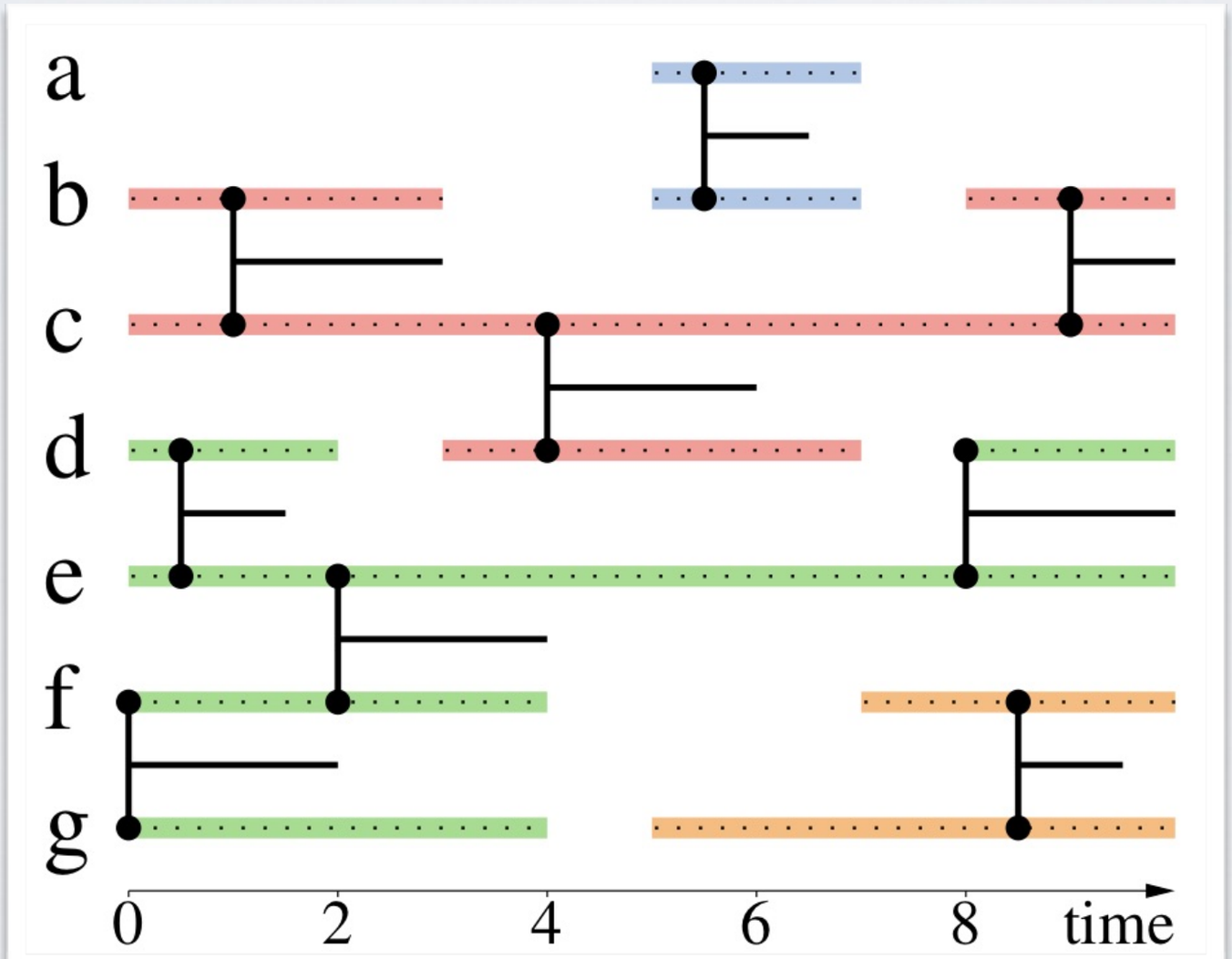


Shortest Fastest path from (1, d) to (9, c) ?

OTHER DEFINITIONS ON STREAM GRAPHS

CONNECTED COMPONENTS

- Weakly connected component:
  - There is at least a non-temporally respecting path



CLOSENESS - BETWEENNESS

$$\mathcal{C}_t(v) = \sum_{u \in V} \int_{\substack{s \in T \\ (s,u) \neq (t,v)}} \frac{1}{c_t(v,(s,u))} \, \mathrm{d}s$$

Shortest path in Static graphs is replaced by a *cost function*, any notion of distance (typically, time to reach)

$$\mathcal{B}(t,v) = \sum_{u \in V, w \in V} \int_{i \in T_u, j \in T_w} \frac{\sigma((i,u),(j,w),(t,v))}{\sigma((i,u),(j,w))} \, \mathrm{d}i \, \mathrm{d}j$$

Proportion of all the shortest fastest paths between all possible (time,node) pairs that go through (t,v)

Latapy, M., Viard, T., & Magnien, C. (2018). Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, *8*(1), 61.

RANDOM MODELS FOR DYNAMIC NETWORKS

RANDOM MODELS

- In many cases, in network analysis, useful to compare a network to a randomized version of it
  - Clustering coefficient, assortativity, modularity, …
- In a static graph, 2 main choices:
  - Keep only the number of edges (ER model)
  - Keep the number of edges and the degree of nodes (Configuration model)
- In dynamic networks, it is more complex…

RANDOM MODELS

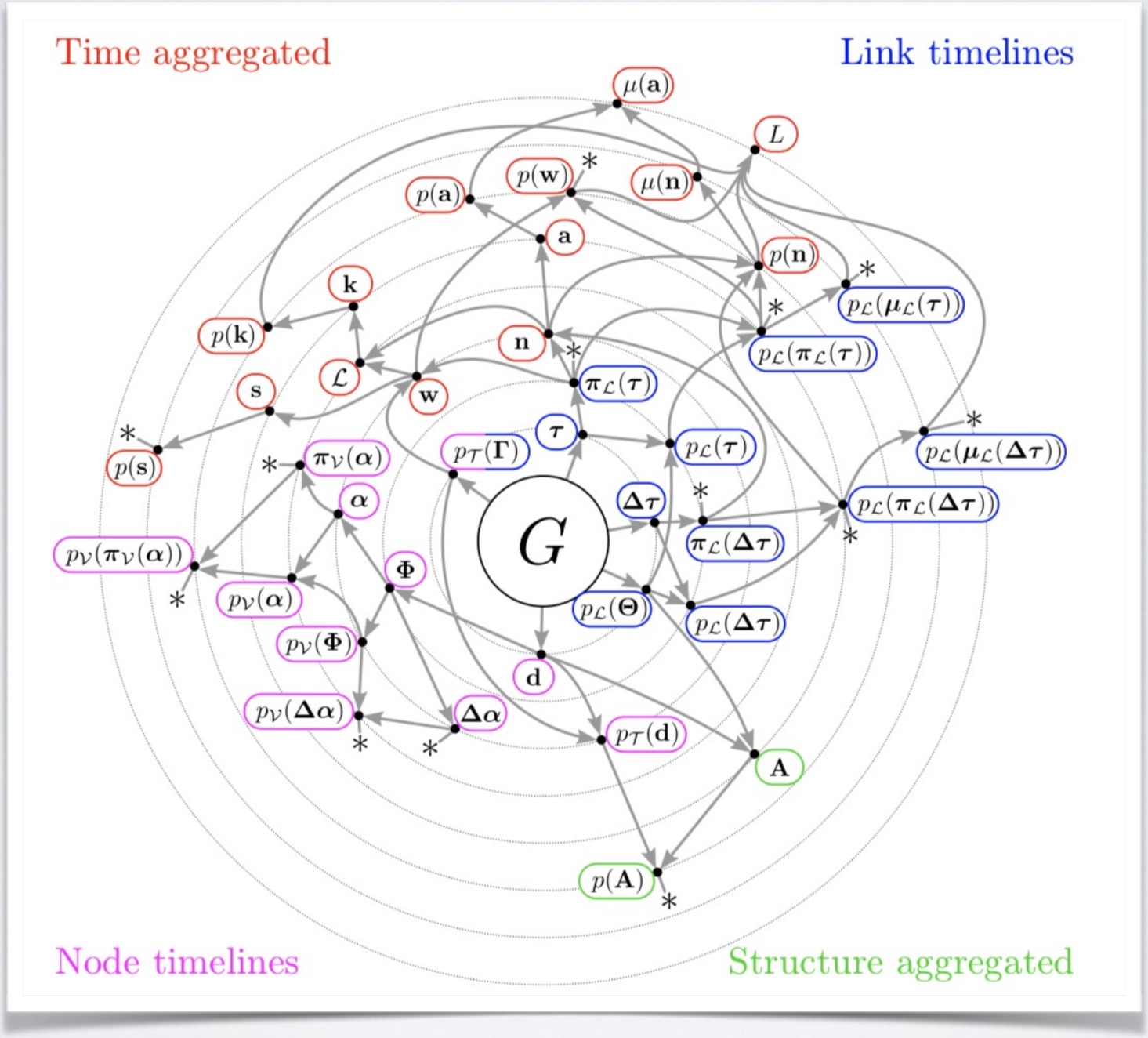
- [Gauvin 2018]

Gauvin, Laetitia, et al. "Randomized reference models for temporal networks." *arXiv preprint arXiv:1806.04032* (2018).

- Four families of shuffling:
  - Snapshot shuffling
    - =>Keep the order of snapshots, randomize network inside snapshot
  - Sequence Shuffling
    - =>Keep each snapshot identical, but switch randomly their order
  - Link Shuffling
    - => Randomize aggregated graph, keep activation times.
    - e.g., pick two node pairs activation time (u1,v1: t0, t1,…), (u2,v2: w0, w1, …) ad switch their activation time.
  - Timeline shuffling
    - => Randomize nodes/edges activation time, conserve the aggregated graph.
    - e.g. pick two edge observations (u1,v1,t1), (u2,v2,t2), switch t1 and t2

- Shufflings can be combined…

RANDOM MODELS



Gauvin, Laetitia, et al. "Randomized reference models for temporal networks." *arXiv preprint arXiv:1806.04032* (2018).

ADM network

with

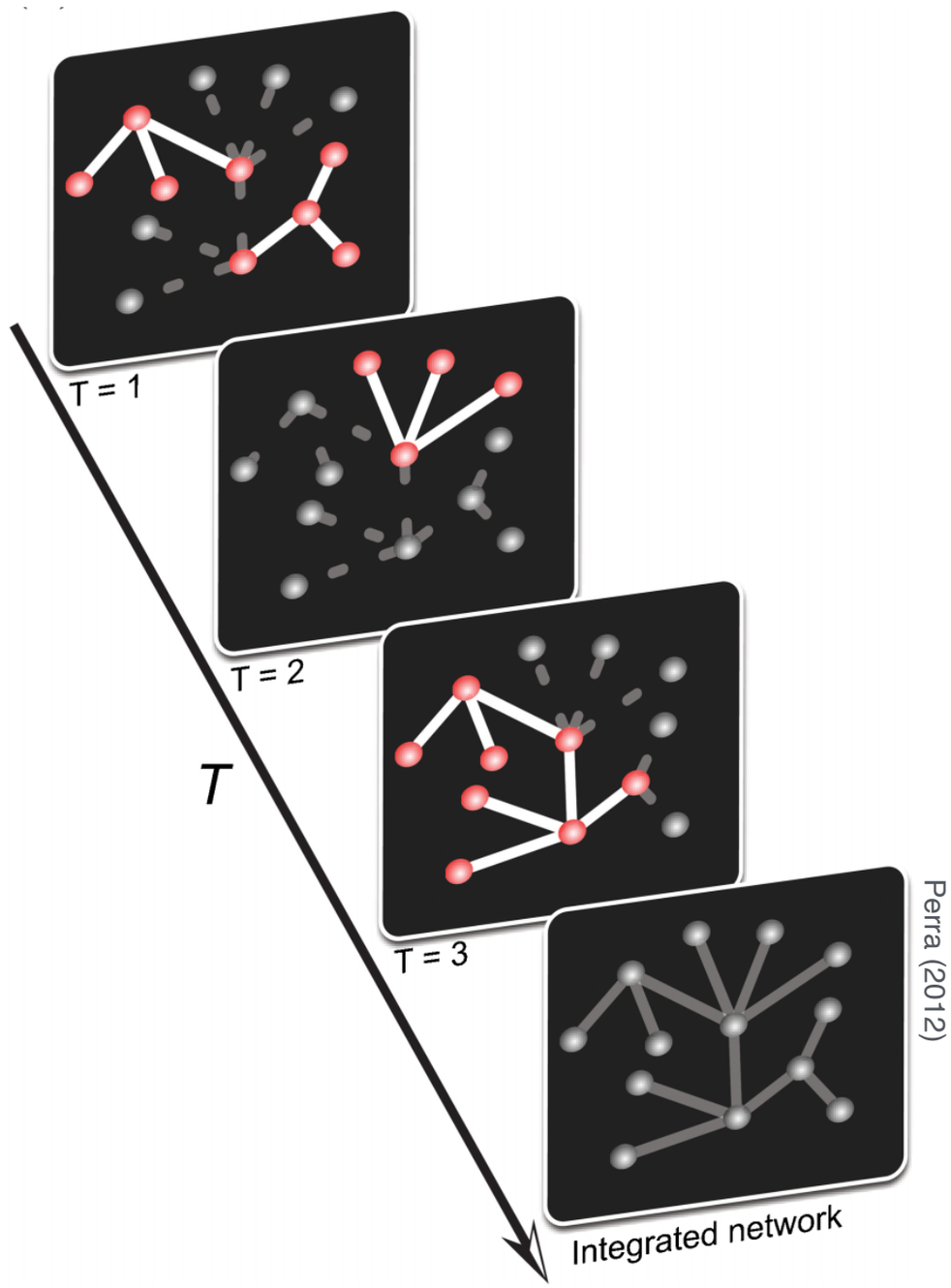
Social mechanisms

Activity driven model of time varying networks

N. Perra, et.al., *Sci. Rep.* **2**, 469 (2012)

Agent based model of temporal interactions

- It is only a general framework where additional mechanisms can be added
- It is capable of simulating dynamical processes co-evolving with the contact dynamics
- It takes a single assumption a priori: agents have different activity potentials



Activity driven model of time varying networks

Definition

- N disconnected nodes, with pre-assigned activity rates:

$$a_i = \eta x_i$$

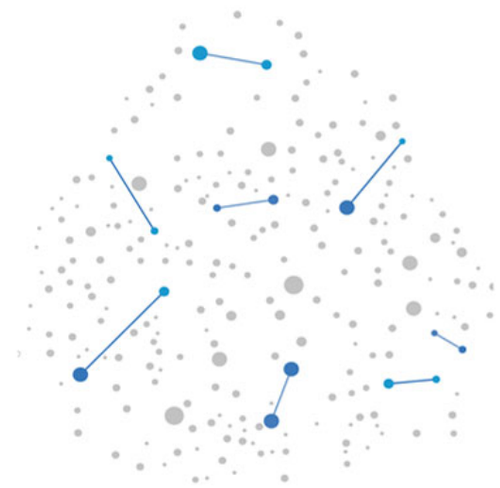
where

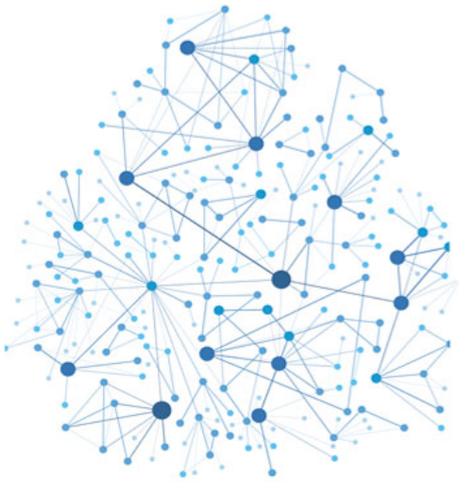
  - x_i is the activity potential of node i - sampled from an arbitrary distribution $F(x)$ and $x_i \in [\varepsilon, 1]$
  - η is a rescaling factor

- Each time step t start with N disconnected nodes:

  1. With probability a_i node i is activated and connect to m other nodes randomly
  2. With probability $1\text{-}a_i$ node i remains inactive (still can receive connections from other active nodes)

- In the end of each time step we delete each link and start the loop over again





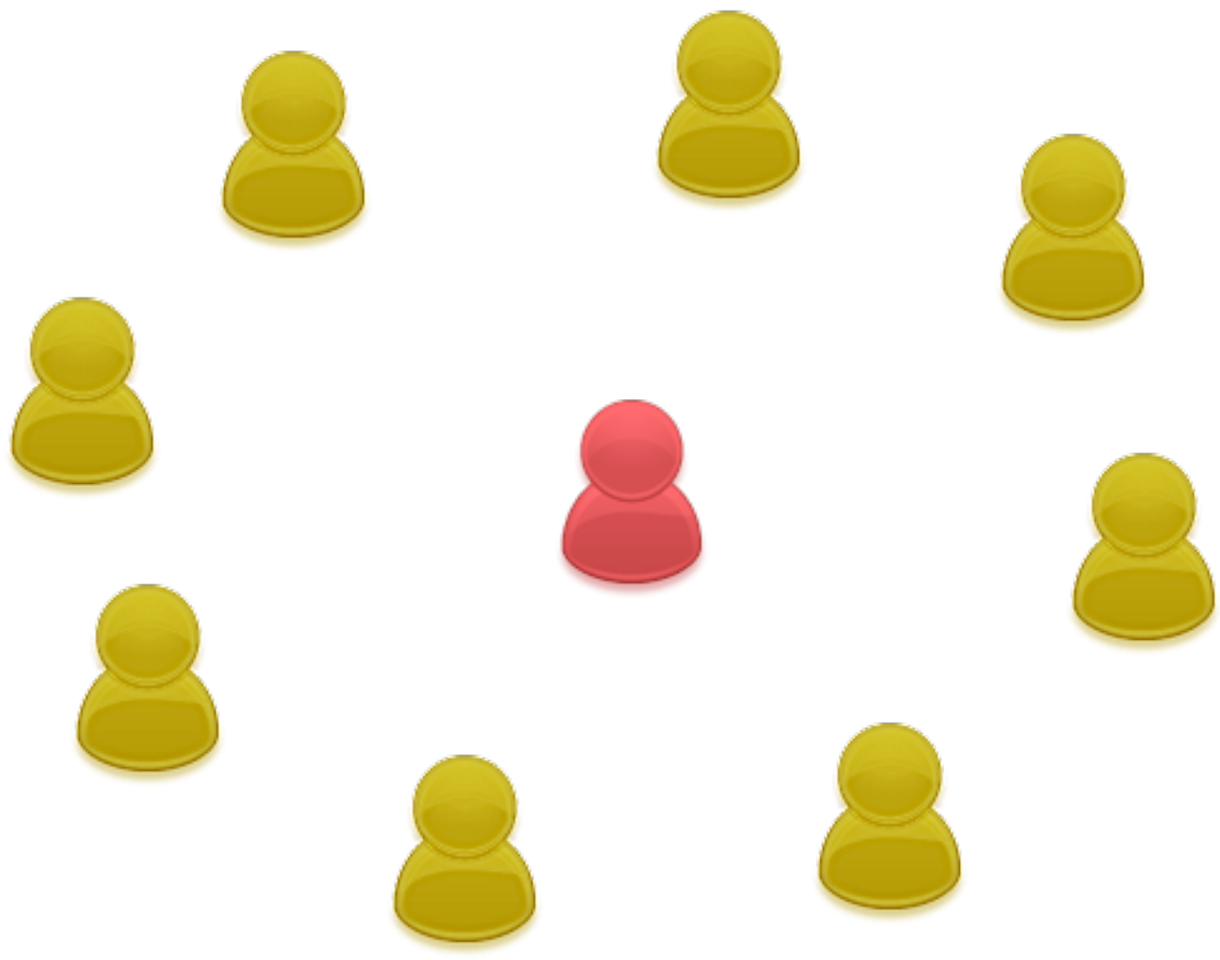
Activity driven model of time varying networks

Features

- The structure of the actual network at each t will be a random network
- The emerging degree distribution of the integrated network will follow the same scaling form as the pre-assigned activity distribution

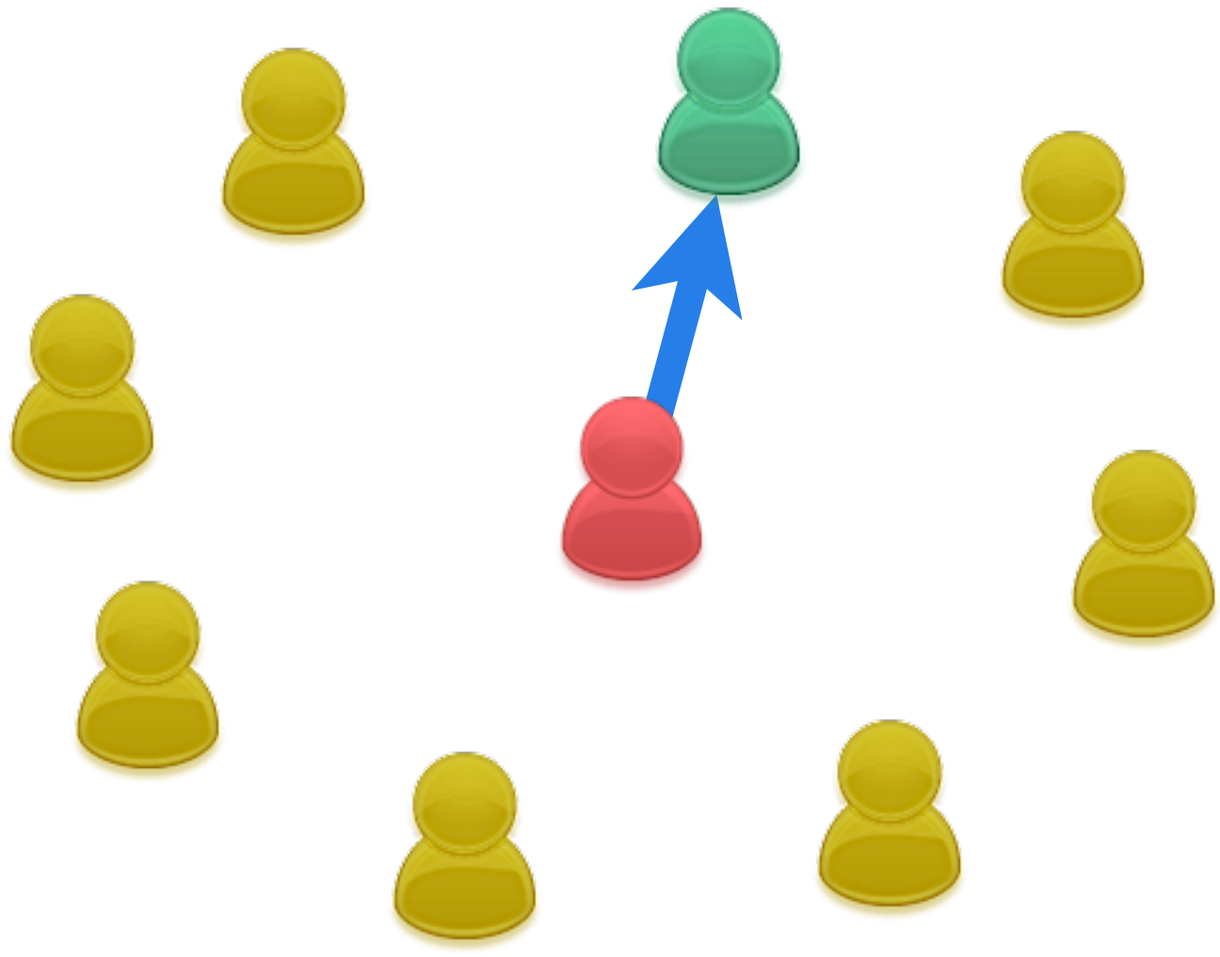
- Real node activity is different…

Egocentric network dynamics



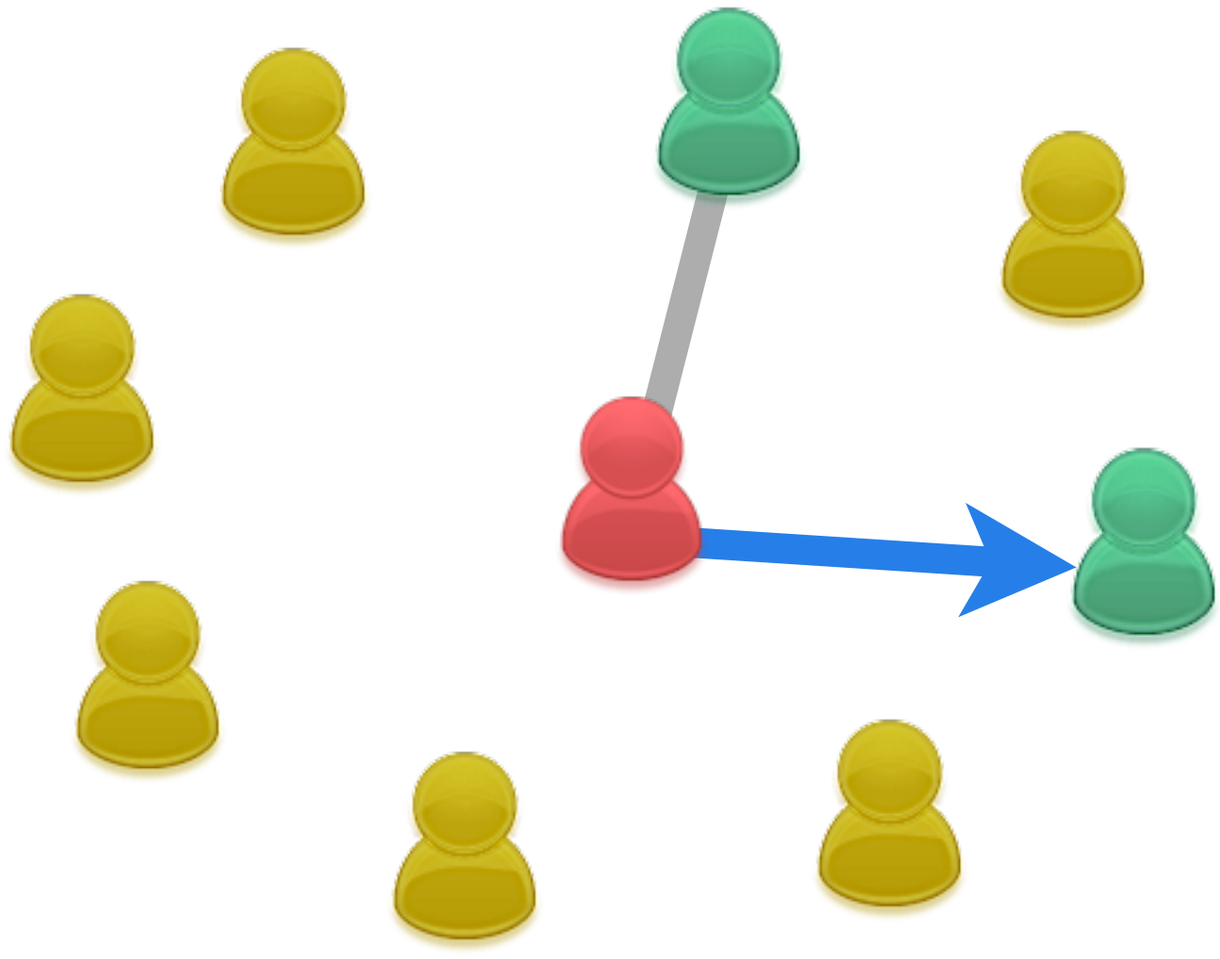
n=0

Egocentric network dynamics



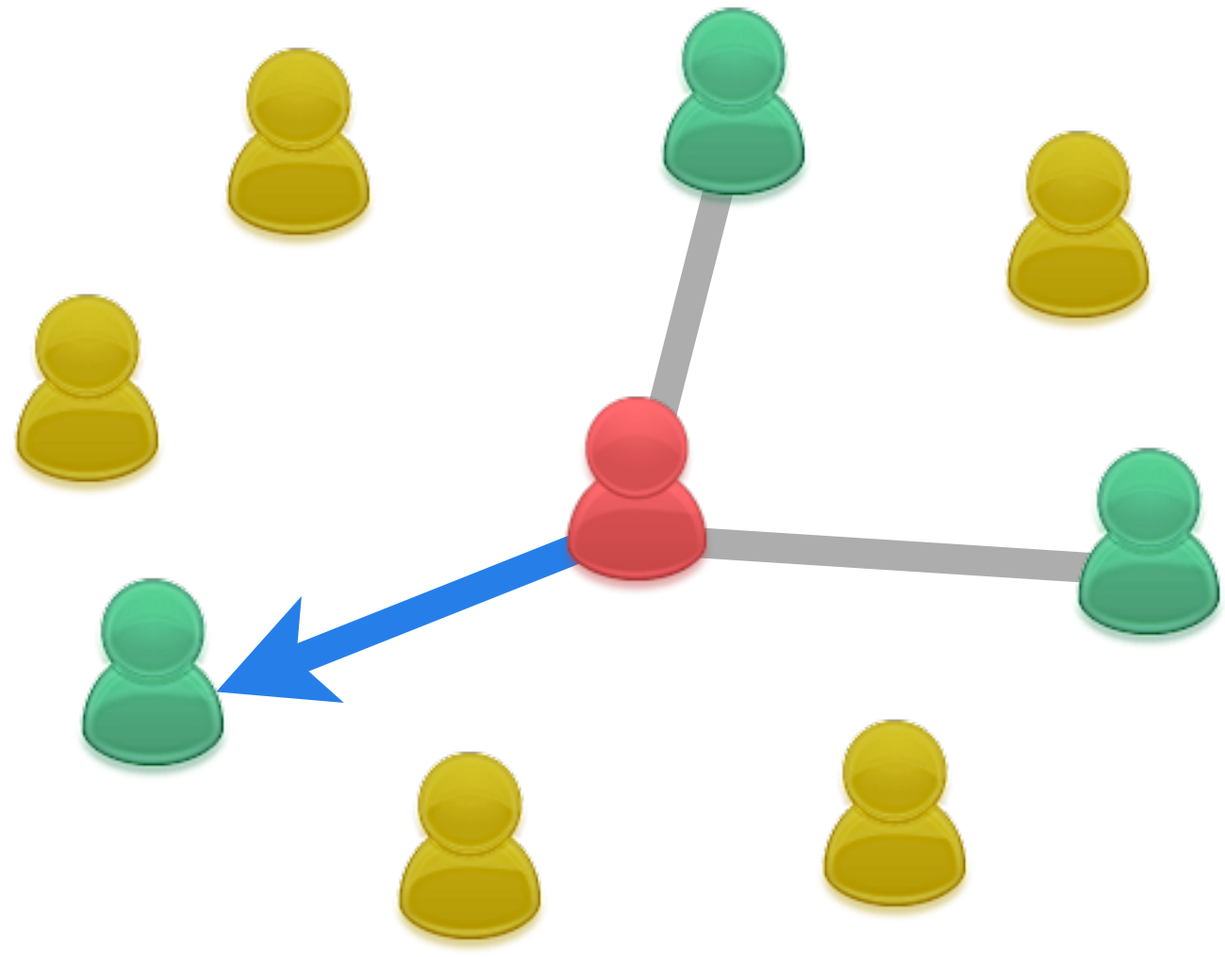
n=1

Egocentric network dynamics



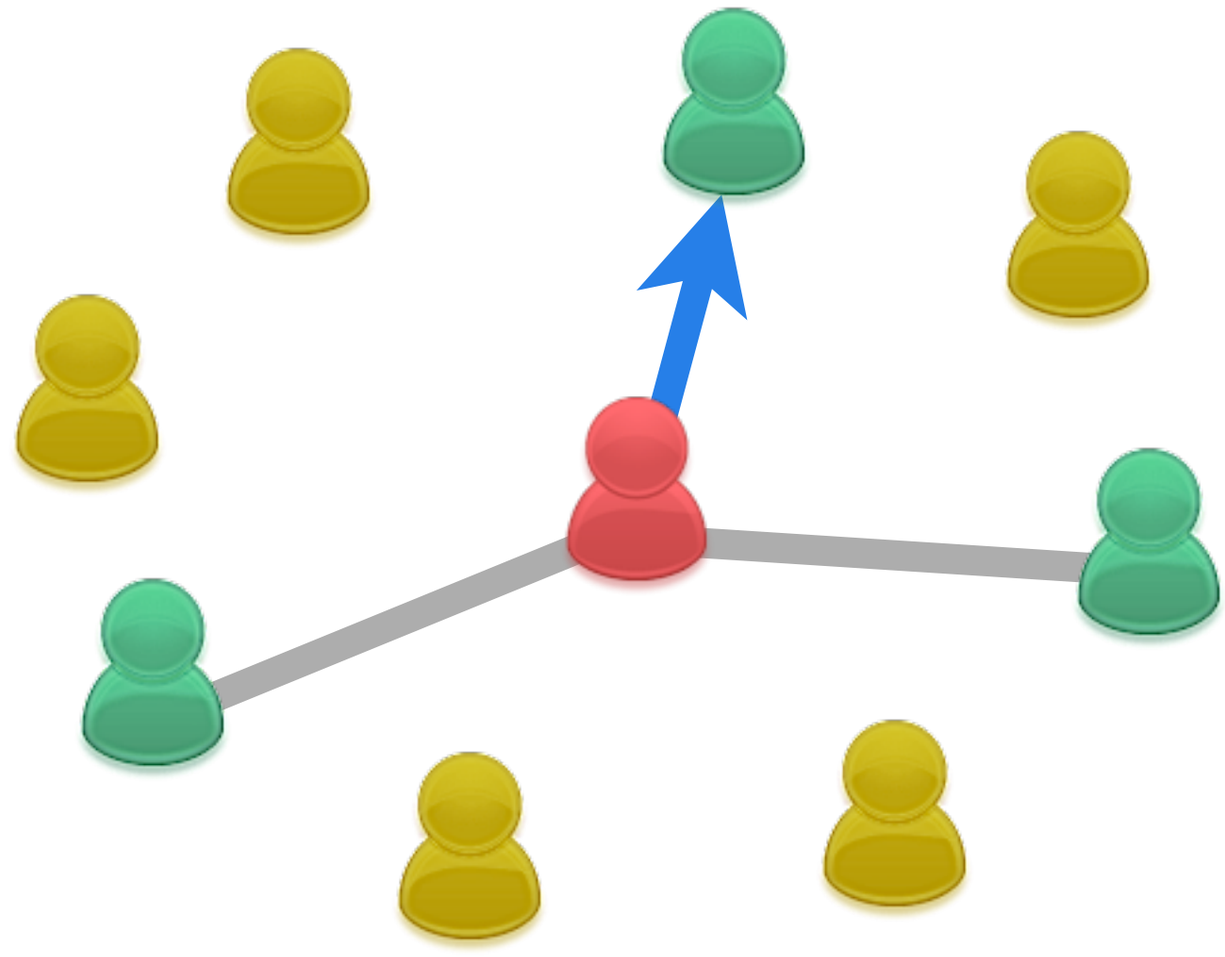
n=2

Egocentric network dynamics



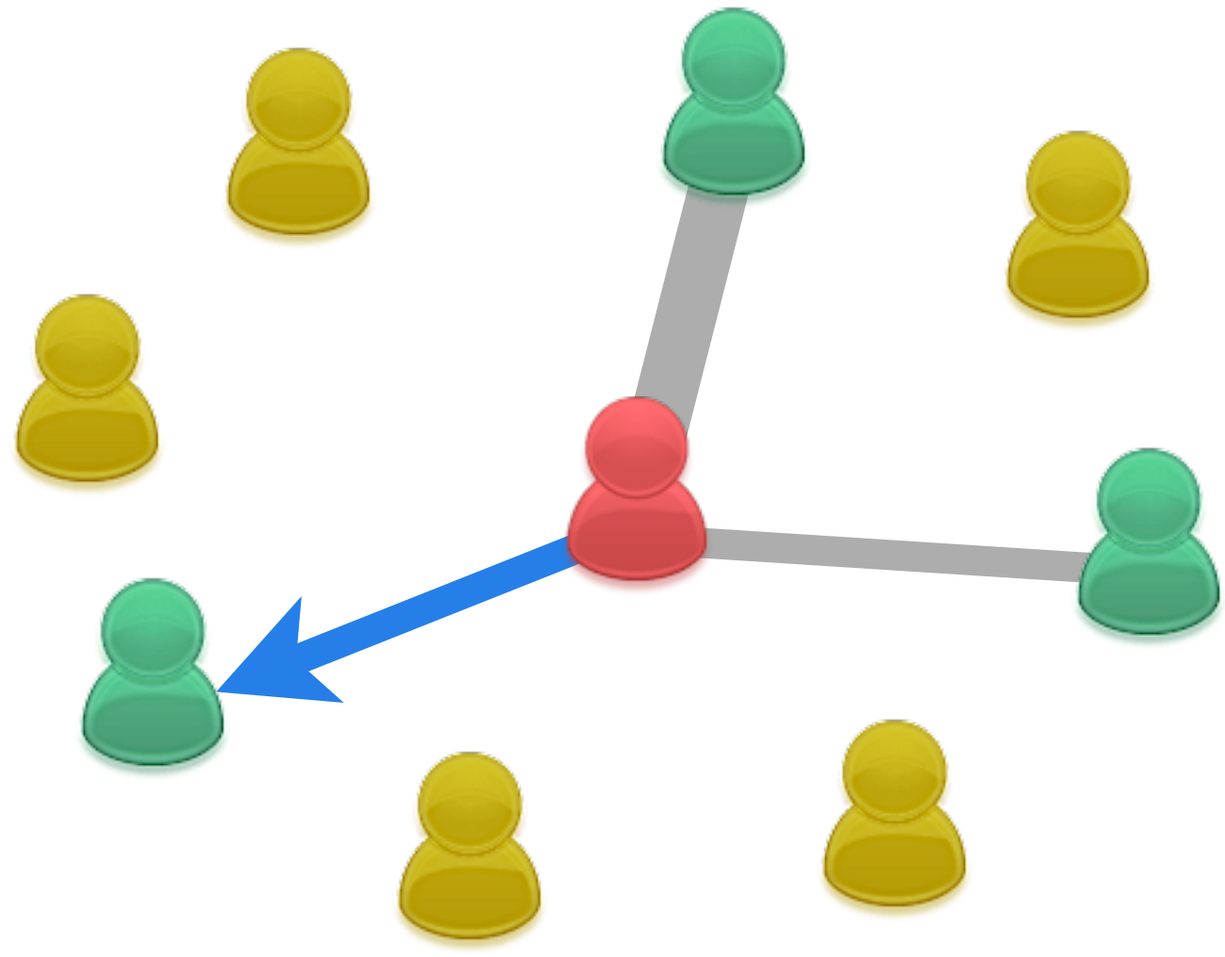
n=3

Egocentric network dynamics



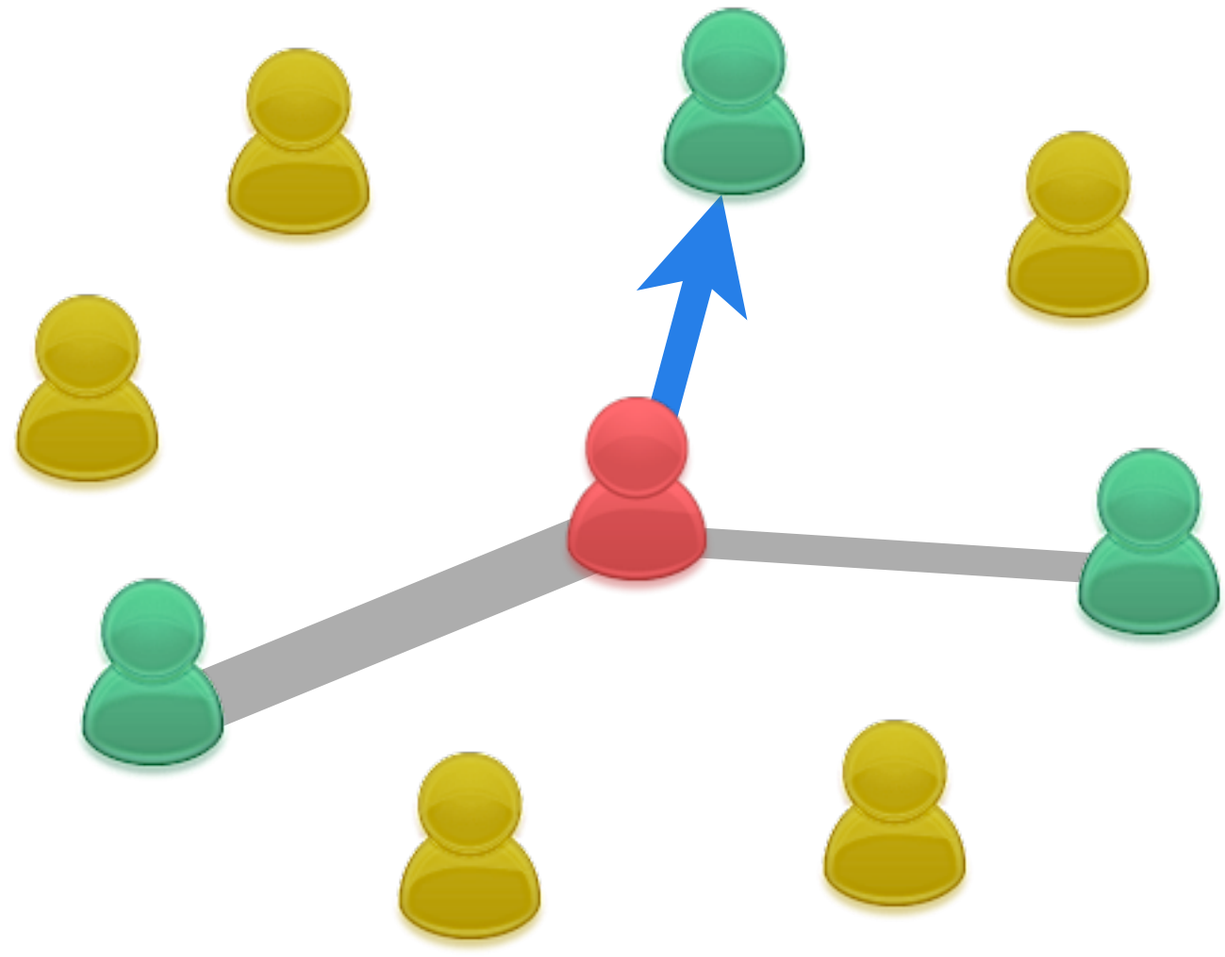
n=3

Egocentric network dynamics



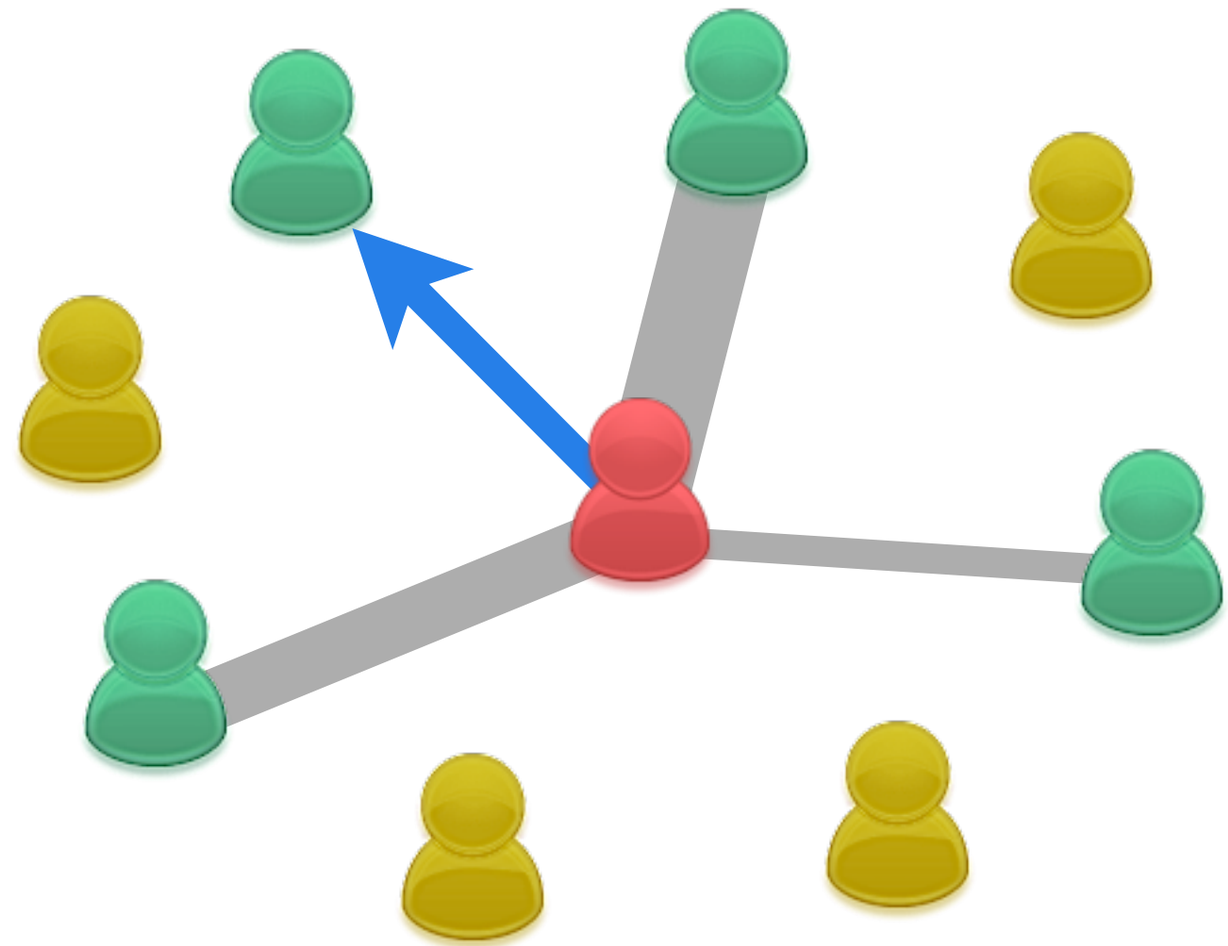
n=3

Egocentric network dynamics



n=3

Egocentric network dynamics



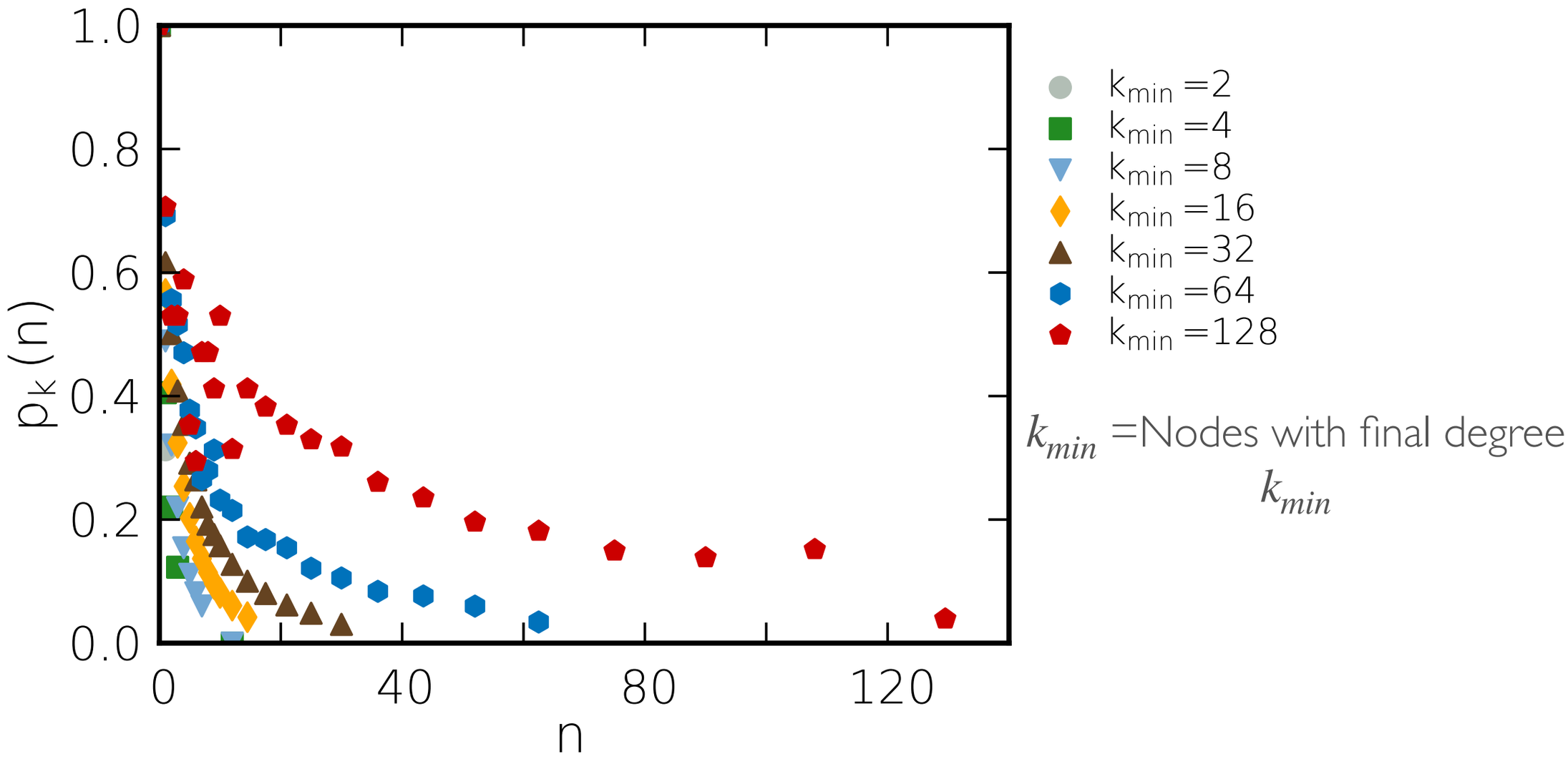
n=4

Egocentric network dynamics with memory

p(n) : probability that the next communication event of an agent with n social ties will occur via the establishment of a new (n + 1) social tie

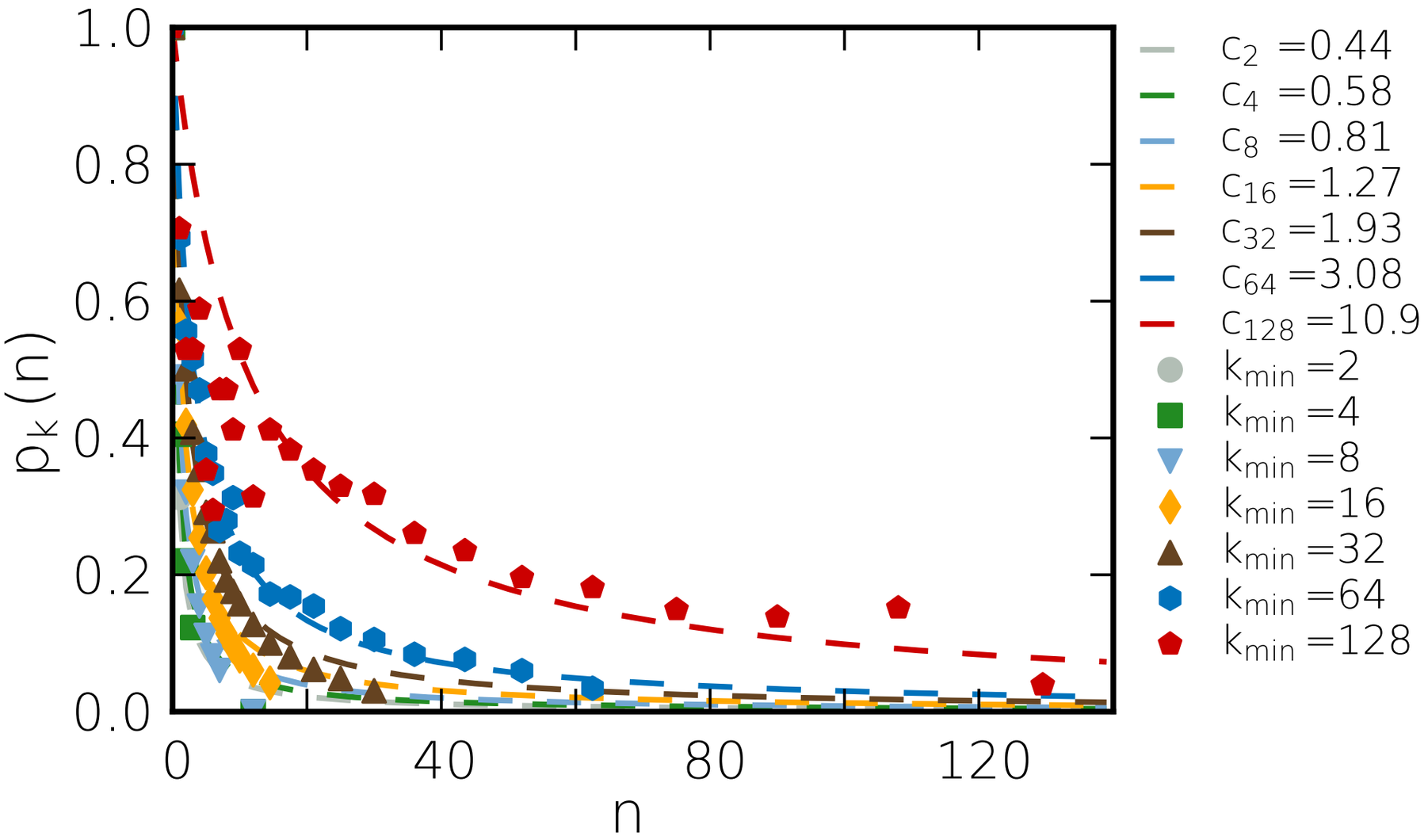
Egocentric network dynamics with memory

p(n) : probability that the next communication event of an agent with n social ties will occur via the establishment of a new (n + 1) social tie



Egocentric network dynamics with memory

p(n) : probability that the next communication event of an agent with n social ties will occur via the establishment of a new (n + 1) social tie

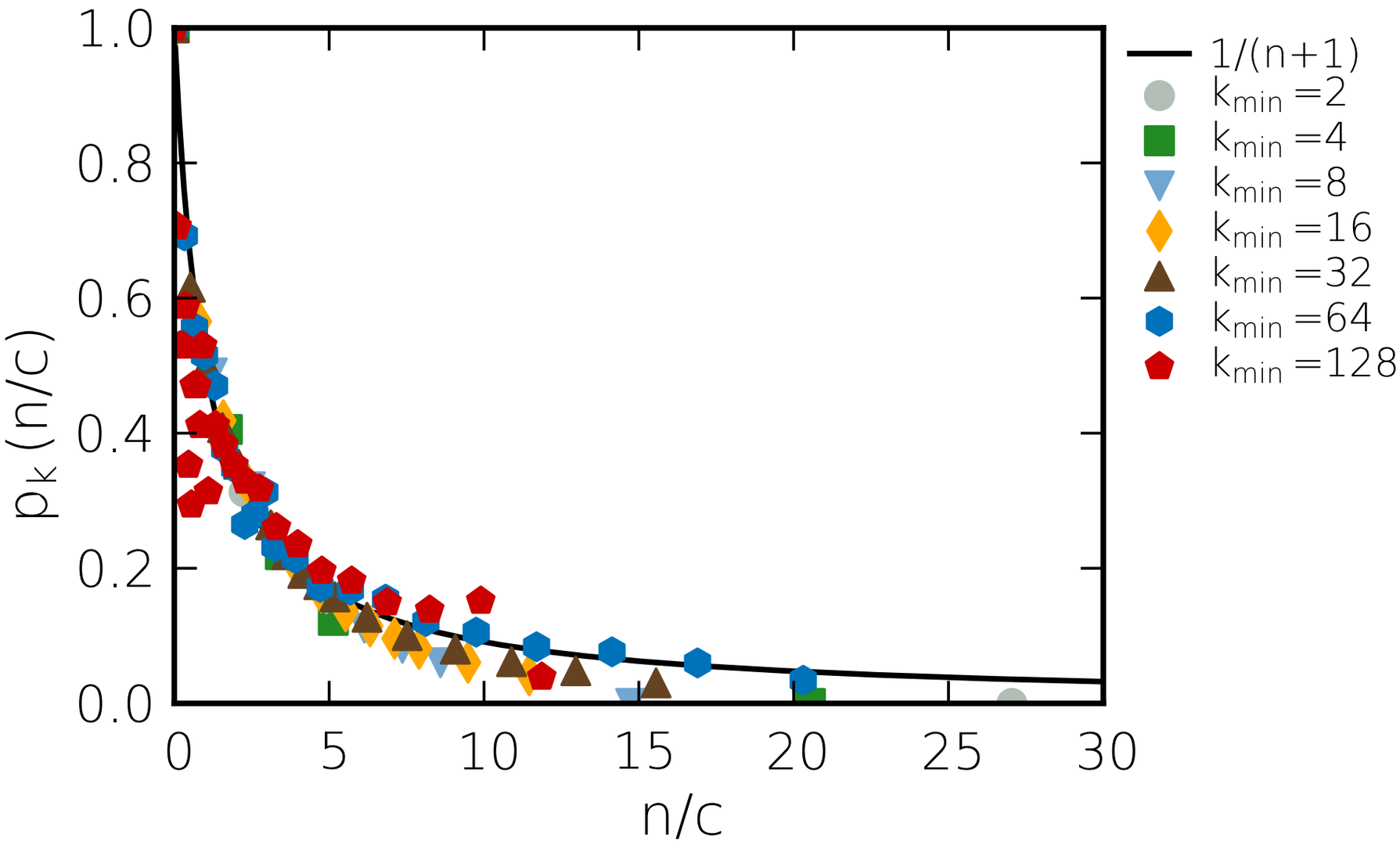


$$p(n) = 1 - \frac{n}{n + c}$$

C an offset constant for each class

Egocentric network dynamics with memory

p(n) : probability that the next communication event of an agent with n social ties will occur via the establishment of a new (n + 1) social tie



$$p(n) = 1 - \frac{n}{n+c} \qquad p_k(n/c) = \frac{1}{n/c+c}$$

Social mechanisms

Activity driven network model

N. Perra, et.al., Sci. Rep. ***2*** *469 (2012)*

- N disconnected nodes with pre-assigned activity:

$$a_i = x_i \eta$$

where the activity potential is sampled from

$$F(x_i) \sim x_i^{-\nu} \text{ where } \quad x_i \in [\epsilon, 1]$$

and η is a rescaling factor.

- In each iteration nodes become active with probability a_i and connect m nodes randomly.

Memory & social reinforcement

M. Karsai, et.al., Sci. Rep. ***4*** *4001 (2014)*

- When a node is active it connects with probability

$$p(n) = c/(n + c)$$

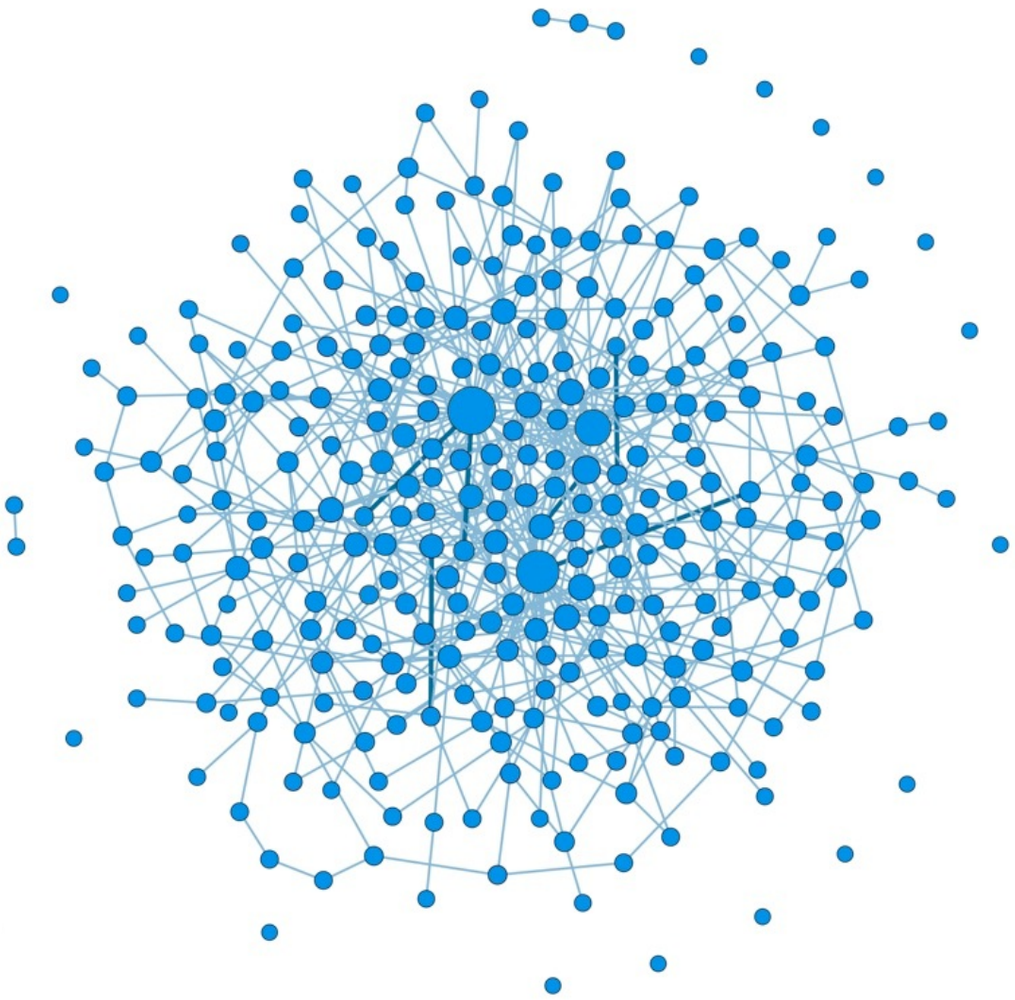
to a random node it has never connected before OR with probability

$$1 - p(n)$$

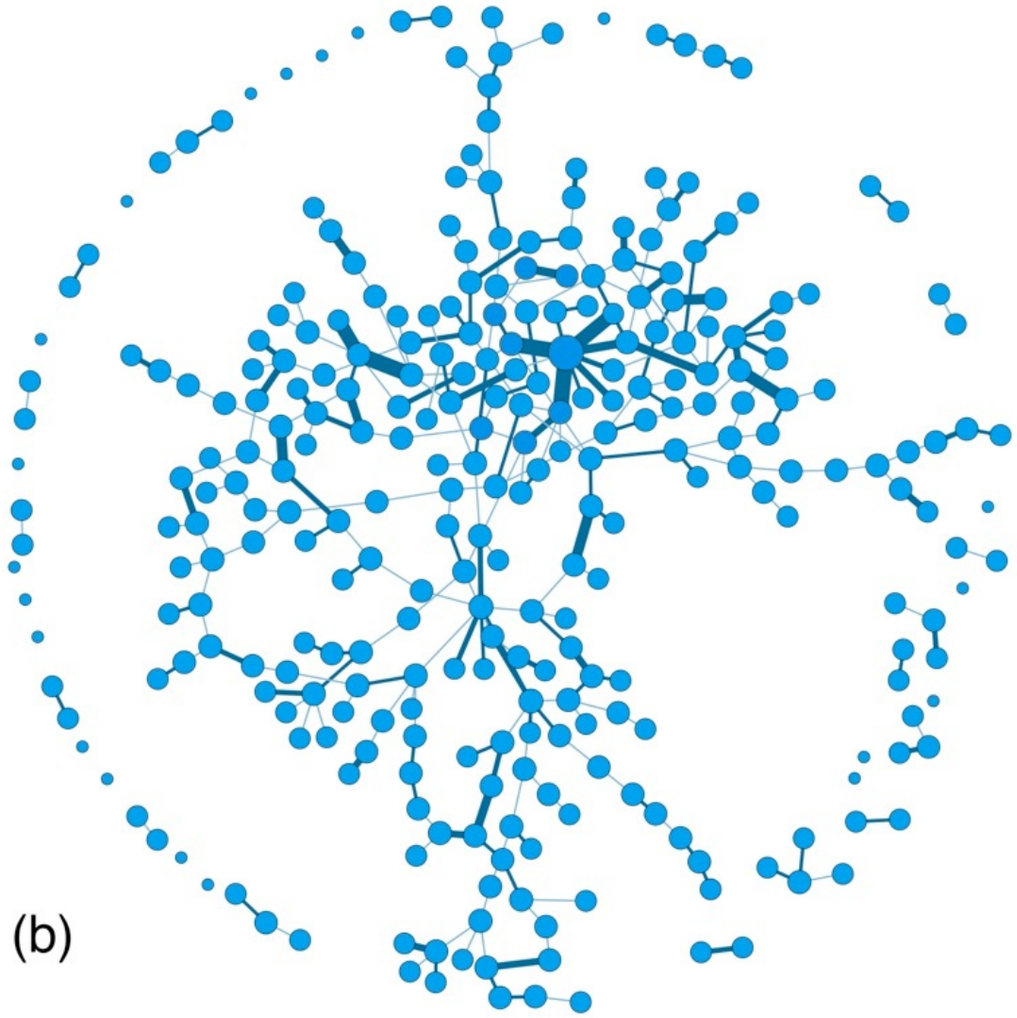
to one of the n node who it has connected earlier

- After each iteration links are deleted but *each node keeps remember to their previously connected egocentric network*
- A node can build a connection by initiating or receiving it

Activity driven network with memory



memoryless process



reinforced process

$$\eta = 1 \qquad \nu = 2.8 \qquad \epsilon = 10^{-3}$$

$$m = 1 \quad \Delta t = 1 \qquad c = 1$$

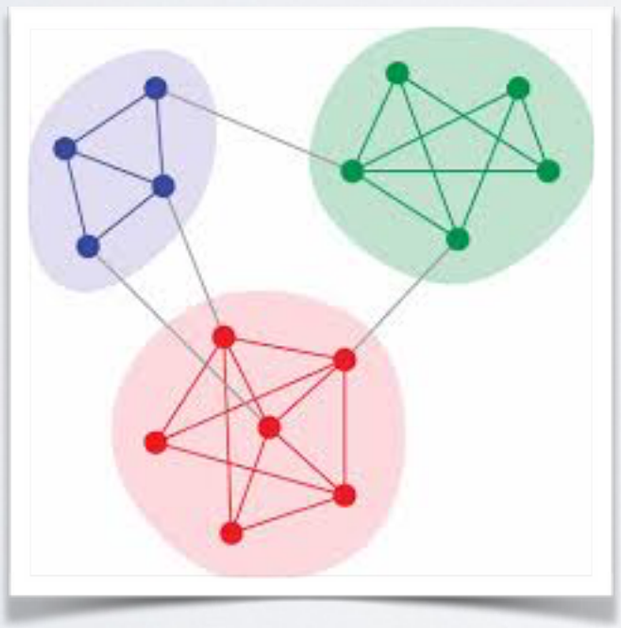
DYNAMIC COMMUNITY DETECTION

Rossetti, Giulio, and Cazabet, Rémy. "Community discovery in dynamic networks: a survey." *ACM Computing Surveys (CSUR)* 51.2 (2018): 35.

COMMUNITY DETECTION

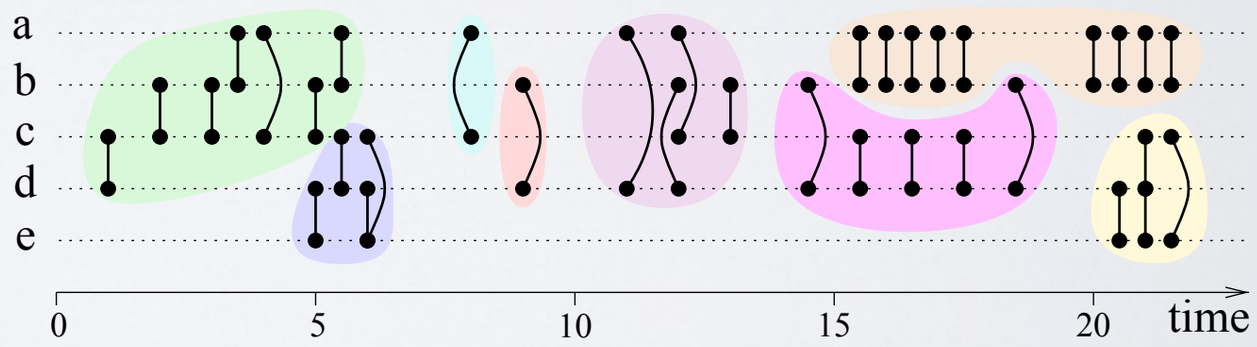
Static networks

Sets of nodes



Dynamic Networks

Sets of periods of nodes

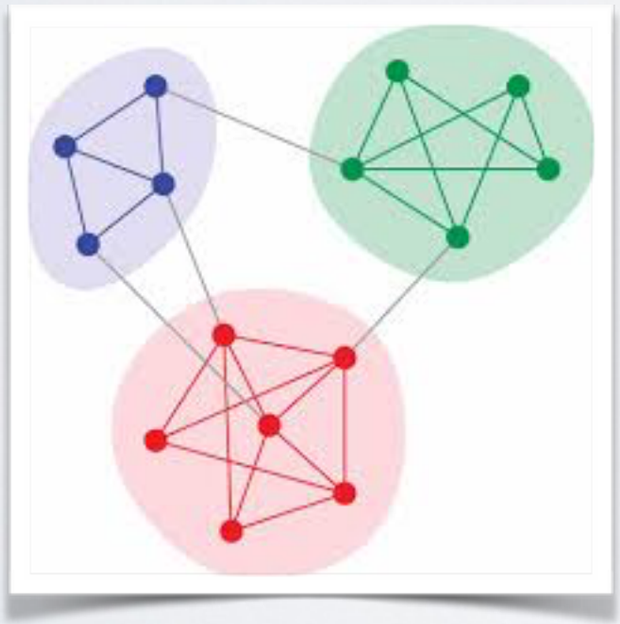


[Viard 2016]

COMMUNITY DETECTION

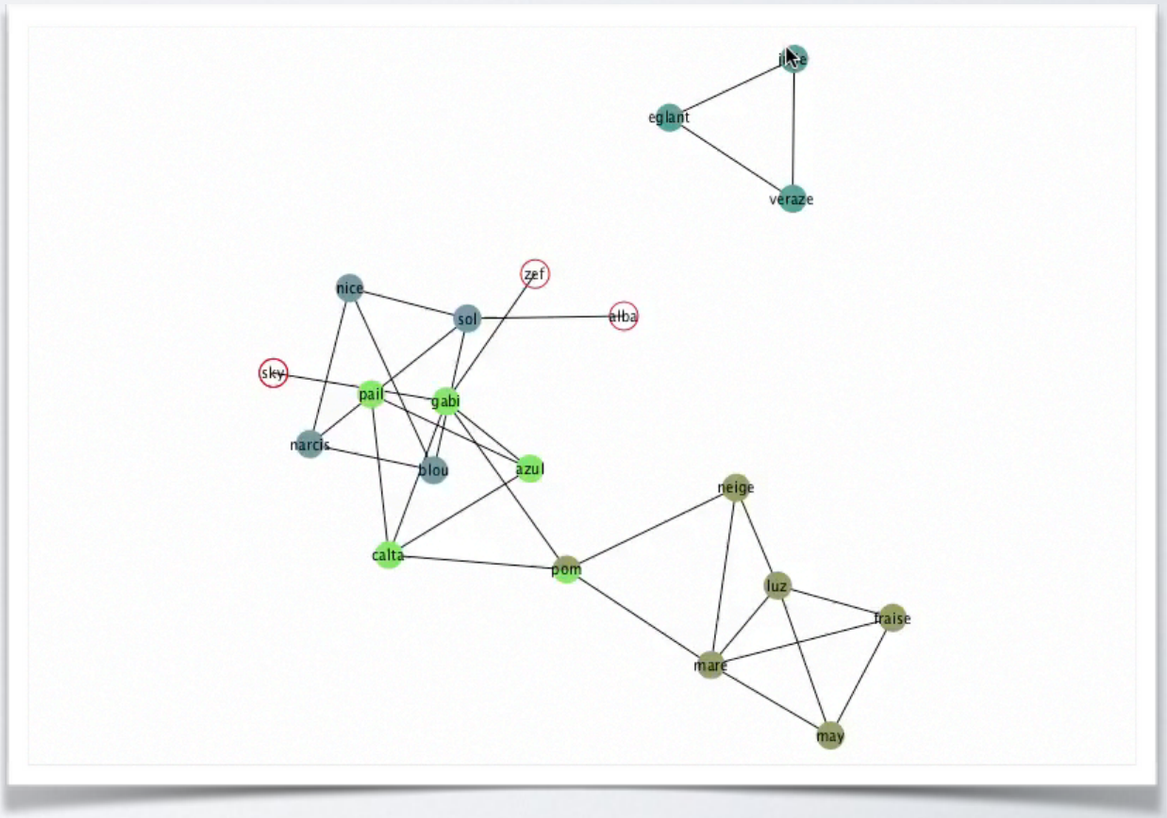
Static networks

Sets of nodes



Dynamic Networks

Sets of periods of nodes

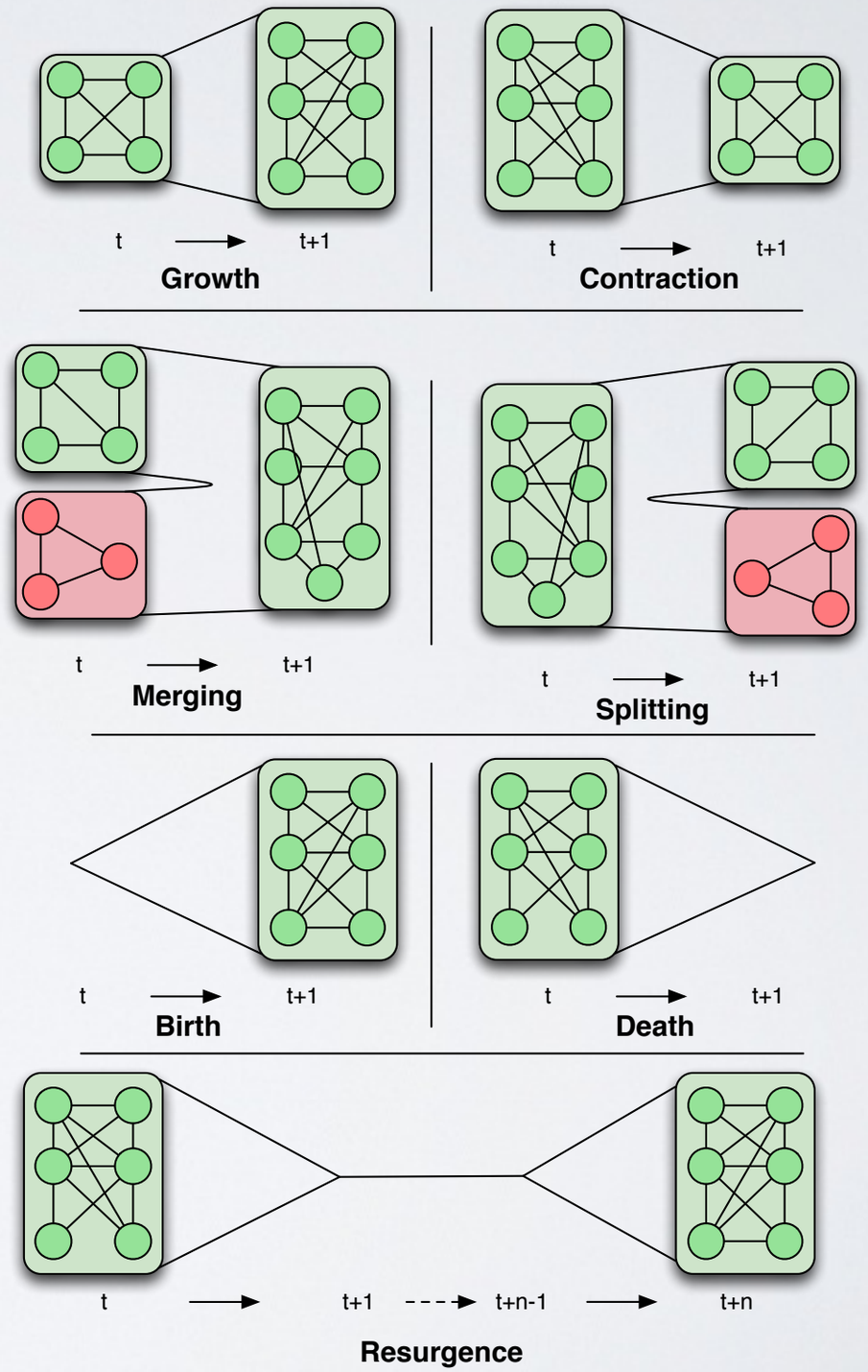


DYNAMIC CD SPECIFICITIES

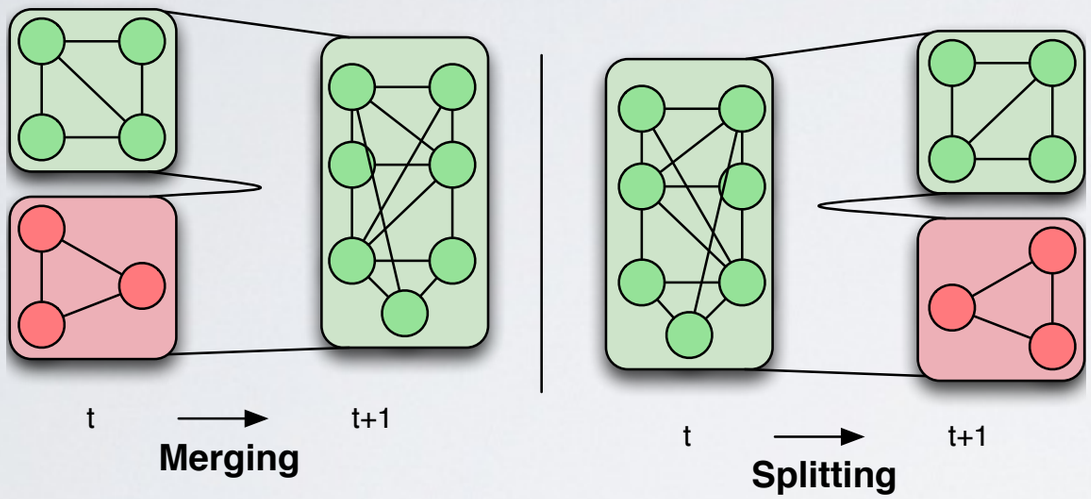
- Dynamic community detection is not a mere iterated static community detection problem
  - Dynamic community detection ?
  - 1) Apply Louvain algorithm at every step
  - 2)Match somehow
  - 3)Problem solved.

COMMUNITY DETECTION

Community events
(or operations)



COMMUNITY DETECTION



Which one persists ?

-Oldest ?

-Most similar ?

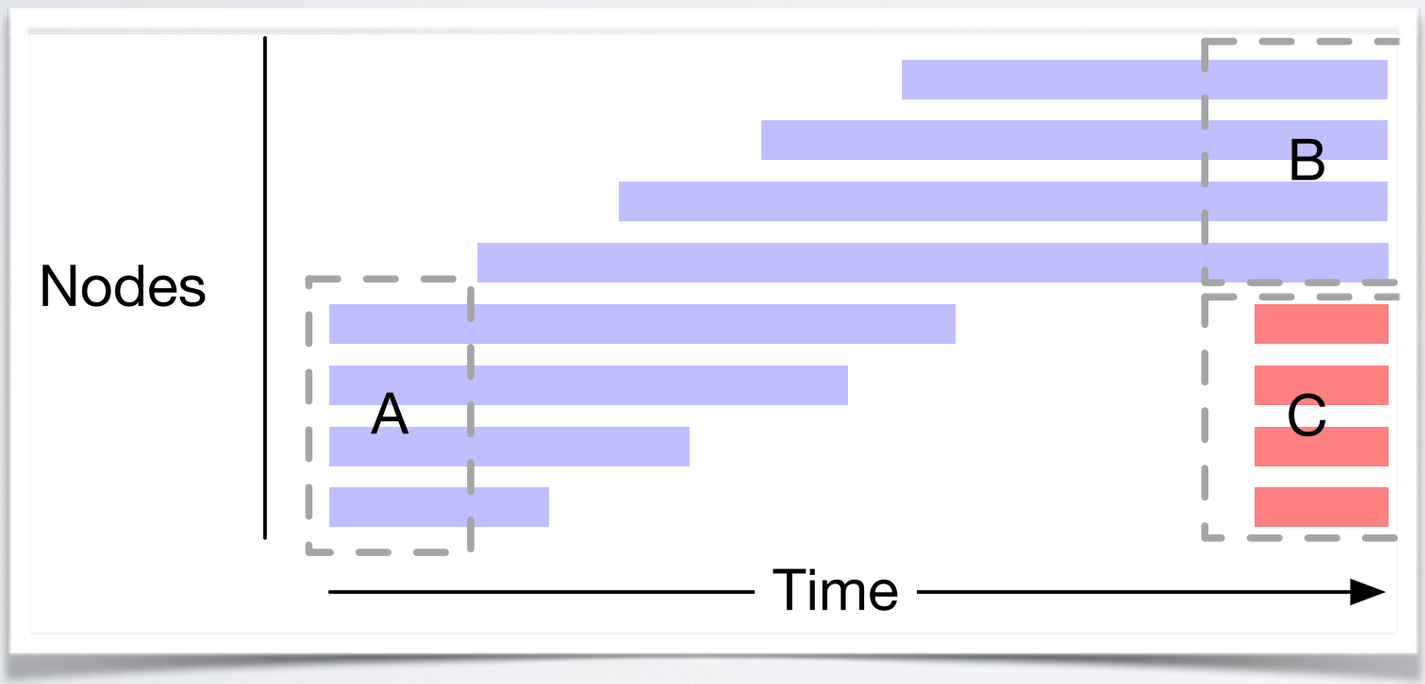
-Larger ?

-...

Community events
(or operations)

IDENTITY PRESERVATION

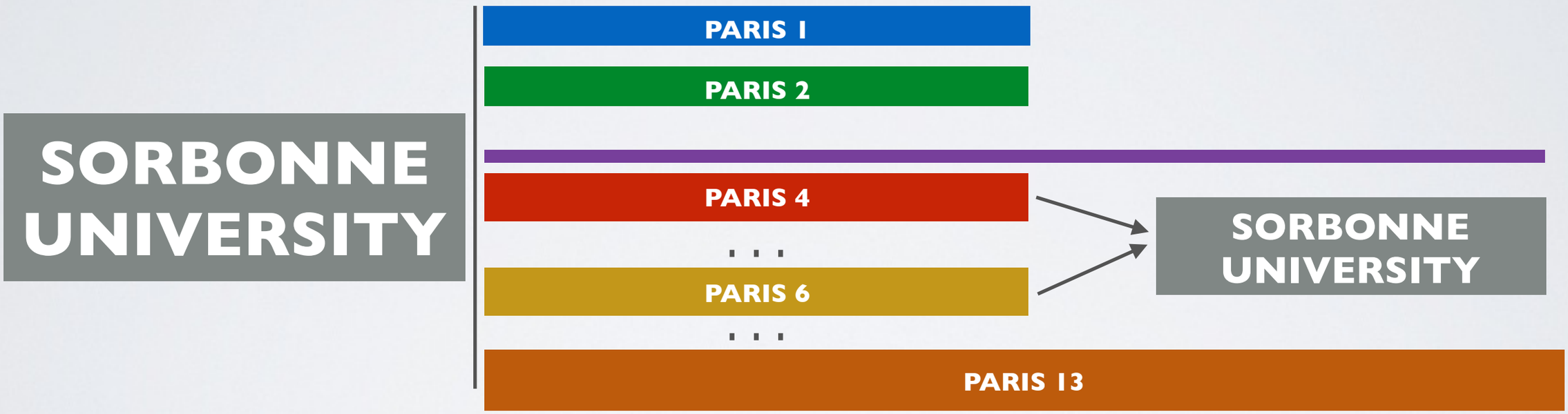
Ship of Theseus [Plutarch., 75]



IDENTITY PRESERVATION

A very concrete problem :

May 68 January 2018



Who gets Marie Curie Nobel Prizes points in the Shanghai University Ranking ?

SMOOTHNESS / STABILITY

- Community smoothness is a fundamental problem

- Algorithms are stochastic / approximations
  - The same algorithm ran twice on the same graph might yield different results
  - Small, local perturbations might generate large, global community changes
    - Shift between different local maximum

SMOOTHNESS / STABILITY

- Let's consider that we have a perfect, deterministic method
  - e.g., always discovering the solution of maximal modularity
  - Each partition might nevertheless be quite different from previous/next one
  - Imagine a borderline situation in which a single edge change makes the community go from 1 to 2 communities, back and forth.

- What we are actually searching is a "simple" (parsimonious) *model*.
  - We want a trade-off between quality and simplicity (smoothness)

SMOOTHNESS / STABILITY

- No Smoothness: Partition at each **t** should be the same as found by a static algorithm.
- Smoothness: Partition at **t** is a trade-off between "good" communities for the graph at **t** and similarity with partitions at different times

COMMUNITY DETECTION

Over 40 methods published

(A) Instant Optimal	**(B)** Temporal Trade-Off	**(C)** Cross-Time
(A1) Iterative, Similarity Based	**(B1)** Update by Global Optimization	**(C1)** Fixed Memberships, Fixed Properties
(A2) Iterative, Core-Node Based	**(B2)** Informed CD by Multi-Objective Optimization	**(C2)** Fixed Memberships, Evolving Properties
(A3) Multi-Step Matching	**(B3)** Update by Set of Rules	**(C3)** Evolving Memberships, Fixed Properties
	(B4) Informed CD by Network Smoothing	**(C4)** Evolving Memberships, Evolving Properties
Clusters at *t* depends **only on the current state** of the network Clusters are **non-temporally smoothed** (Communities **labels**, however, can be smoothed)	Clusters at *t* depends **on current and past states** of the network Clusters are **incrementally temporally smoothed**	Clusters at *t* depends on **both past and future** states of the network Clusters are **Completely temporally smoothed**

CATEGORIES

- Instant optimal:
  - Work only with snapshots
  - No partition smoothing
  - Labels can be smoothed
  - Easy to parallelize

  - =>Best suited for large networks with few steps of evolution (<100?)

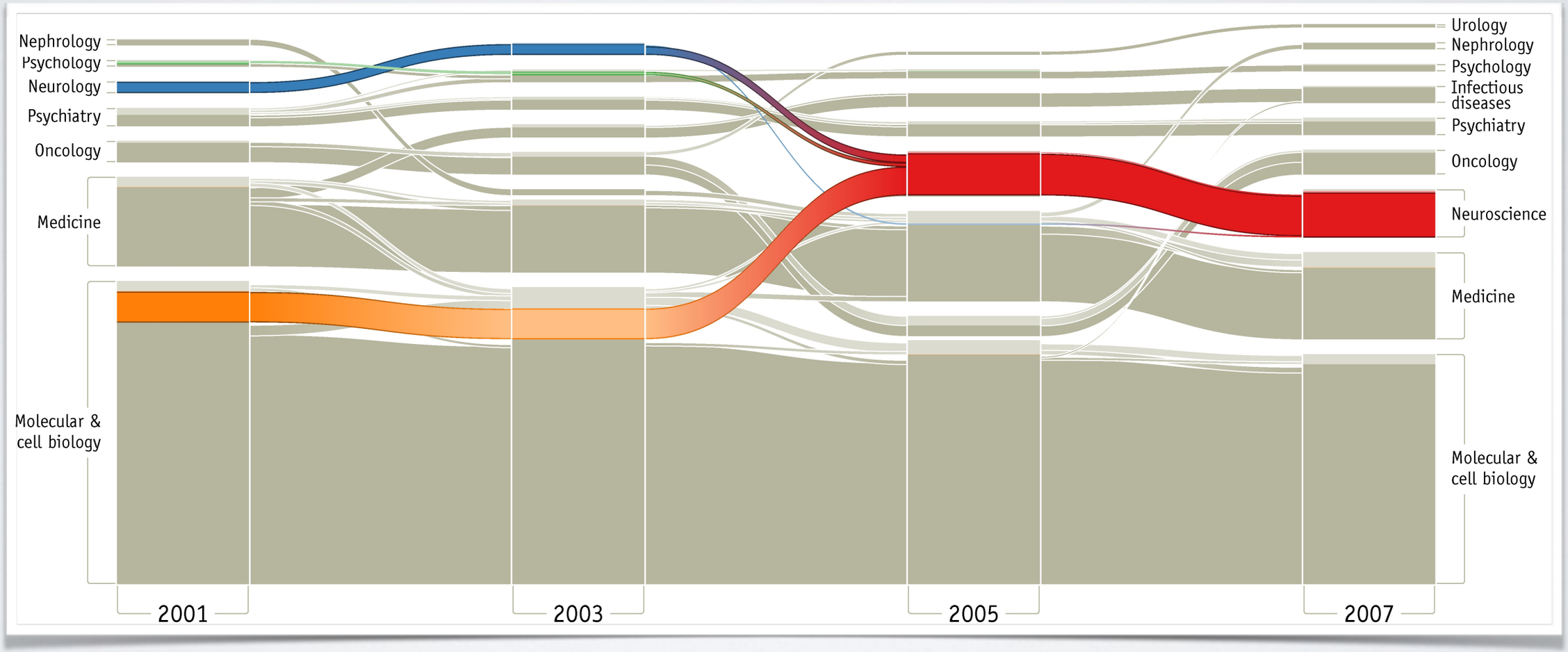
CATEGORIES

- Temporal trade-off
  - Works with any type of temporality (in theory)
  - Cannot be parallelized (iterative)
  - => Best suited for real-time analysis / tasks

- Cross-Time
  - Works with any type of temporality (in theory)
  - Requires to know the whole evolution in advance
  - => Not suited for real-time analysis, potentially the best smoothed (a posteriori interpretation)

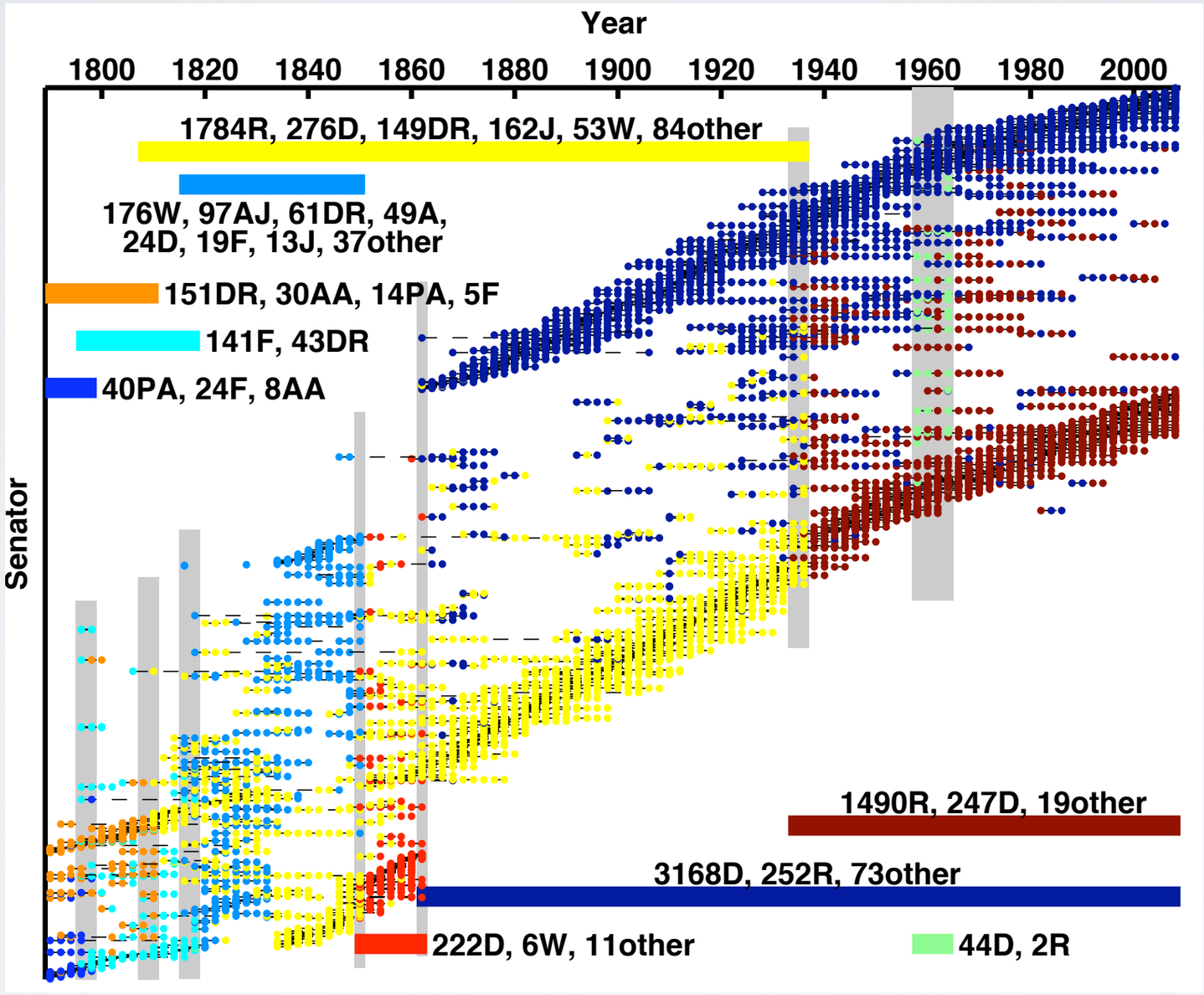
COMMUNITY DETECTION

Some examples of applications



Rosvall et al. 2010

COMMUNITY DETECTION

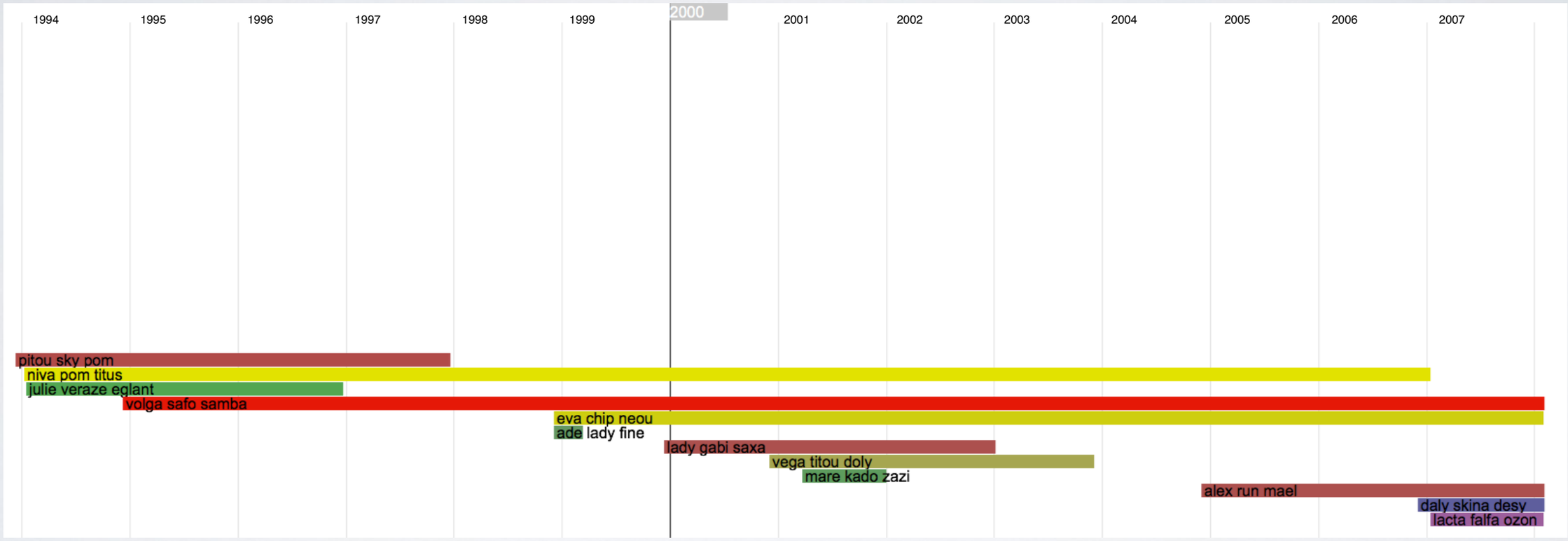


R : Républicains

D : Démocrates

Mucha et al. 2010

COMMUNITY DETECTION



EXAMPLE OF METHODS

EXAMPLE OF METHODS

Same definition of static communities: modularity

- **No-smoothing** (Adapted from [Greene et al.])
- **Implicit Global** (Aynaud et al.)
- **DYNAMO** (Implicit local, Zhuang et al. 2017)
- **DYNMOGA** (Explicit Smoothing, Folino et al 2010)
- **Transversal Network** (Mucha et al.)
- **Label Smoothing** (Falkowski et al.)

EXAMPLE OF METHODS

- **No-smoothing**
  - Communities are detected at every step using a static algorithm (e.g. Louvain Algorithm)
  - Similarities are computed between communities in consecutive steps (at t and t+1 (e.g., Jaccard index)) $J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$
  - Most similar communities are matched between t and t+1

Greene, Derek, Donal Doyle, and Padraig Cunningham. "Tracking the evolution of communities in dynamic social networks." *2010 international conference on advances in social networks analysis and mining*. IEEE, 2010.

EXAMPLE OF METHODS

- **Label-smoothing**
  - Communities are detected at every step using a static algorithm (e.g. Louvain Algorithm)
  - Similarities are computed between all communities in all steps(e.g., Jaccard index))
  - A community network is generated, in which nodes are communities and edges are weighted by similarity
  - A community detection algorithm (e.g., Louvain) is applied on the community network to find dynamic communities.

Falkowski, Tanja, Jörg Bartelheimer, and Myra Spiliopoulou. "Community dynamics mining." *ECIS*. 2006.

EXAMPLE OF METHODS

- **Implicit Global**
  - Communities are detected in the first step using the Louvain algorithm
  - At the next step, the Louvain algorithm is initialized with previous communities as seeds (instead of each node in its own community)
  - =>tend to stay around the same local maximum

Aynaud, Thomas, and Jean-Loup Guillaume. "Static community detection algorithms for evolving networks." *8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*. IEEE, 2010.

EXAMPLE OF METHODS

- **DYNAMO (not working with snaphsots) (implicit local)**
  - Communities are detected in the first step using a static algorithm
  - At each network modification, local rules are applied to preserve a high modularity, without re-running a global optimization

Zhuang, Di, J. Morris Chang, and Mingchen Li. "DynaMo: Dynamic Modularity-based Community Detection in Evolving Social Networks." *arXiv preprint arXiv:1709.08350* (2017).

EXAMPLE OF METHODS

- **DYNMOGA (explicit global)**
  - Communities are detected on the first snapshot using a static algorithm
  - For snapshot t+1, a genetic algorithm is used to solve a multi-objective optimization problem:

- Optimize a partition quality function (e.g., modularity) (*SC, Snapshot Cost)*
- Optimize the similarity to the previous partition (e.g., Normalized Mutual Information) (*TC, T*emporal Cost)
- A parameter α allows to tune the importance of both aspects

$$cost = \alpha SC + (1 - \alpha)TC$$

Folino, Francesco, and Clara Pizzuti. "A multiobjective and evolutionary clustering method for dynamic networks." *2010 International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 2010.

EXAMPLE OF METHODS

- **Transversal Network (cross-time)**
  - A transversal network is built: nodes are couples (nodes, time), edges link the same node in adjacent snapshots
  - A community detection algorithm is run on this transversal network
    - (Note: modified Modularity to avoid overestimating expected edges between nodes in different time steps, i.e., custom random graph)

Mucha, Peter J., et al. "Community structure in time-dependent, multiscale, and multiplex networks." *science* 328.5980 (2010): 876-878.

DCD EVALUATION

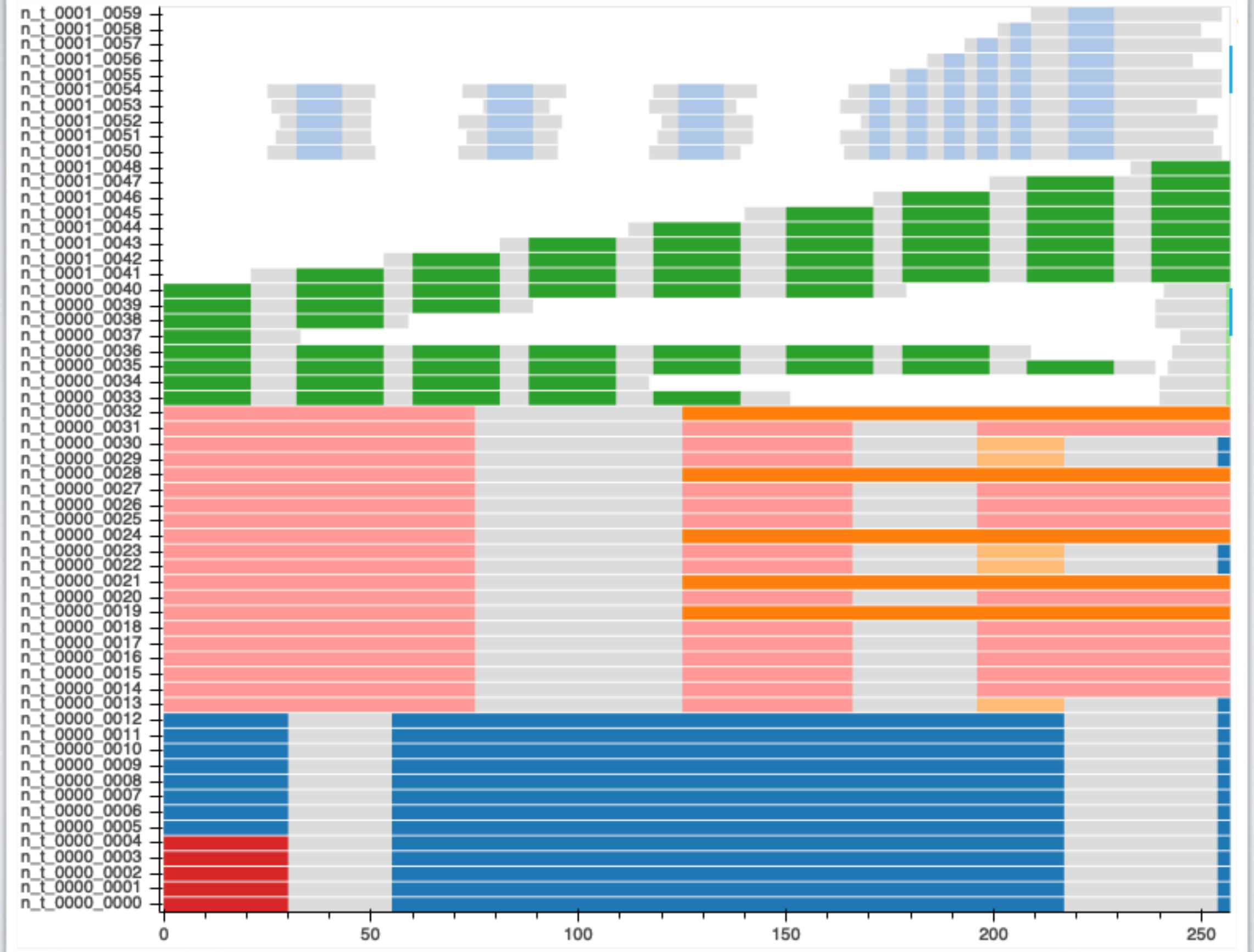
DCD EVALUATION

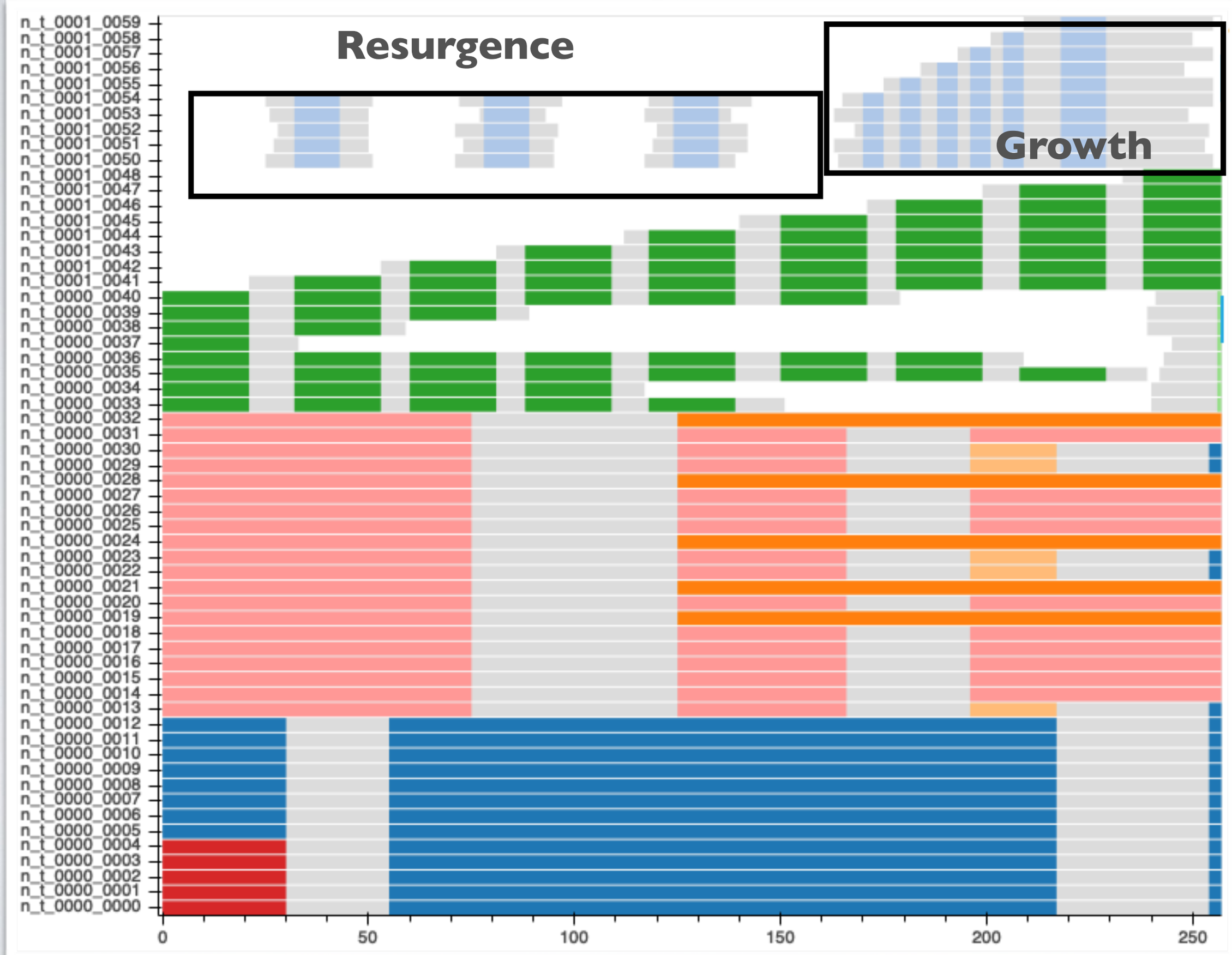
- Tests on synthetic networks
  - We know what we want to find
  - We run algorithms and check the results
- Tests on real networks
  - Start from a real dataset
  - Transform into an appropriate dynamic network (if needed)
  - Run algorithms and try to interpret results

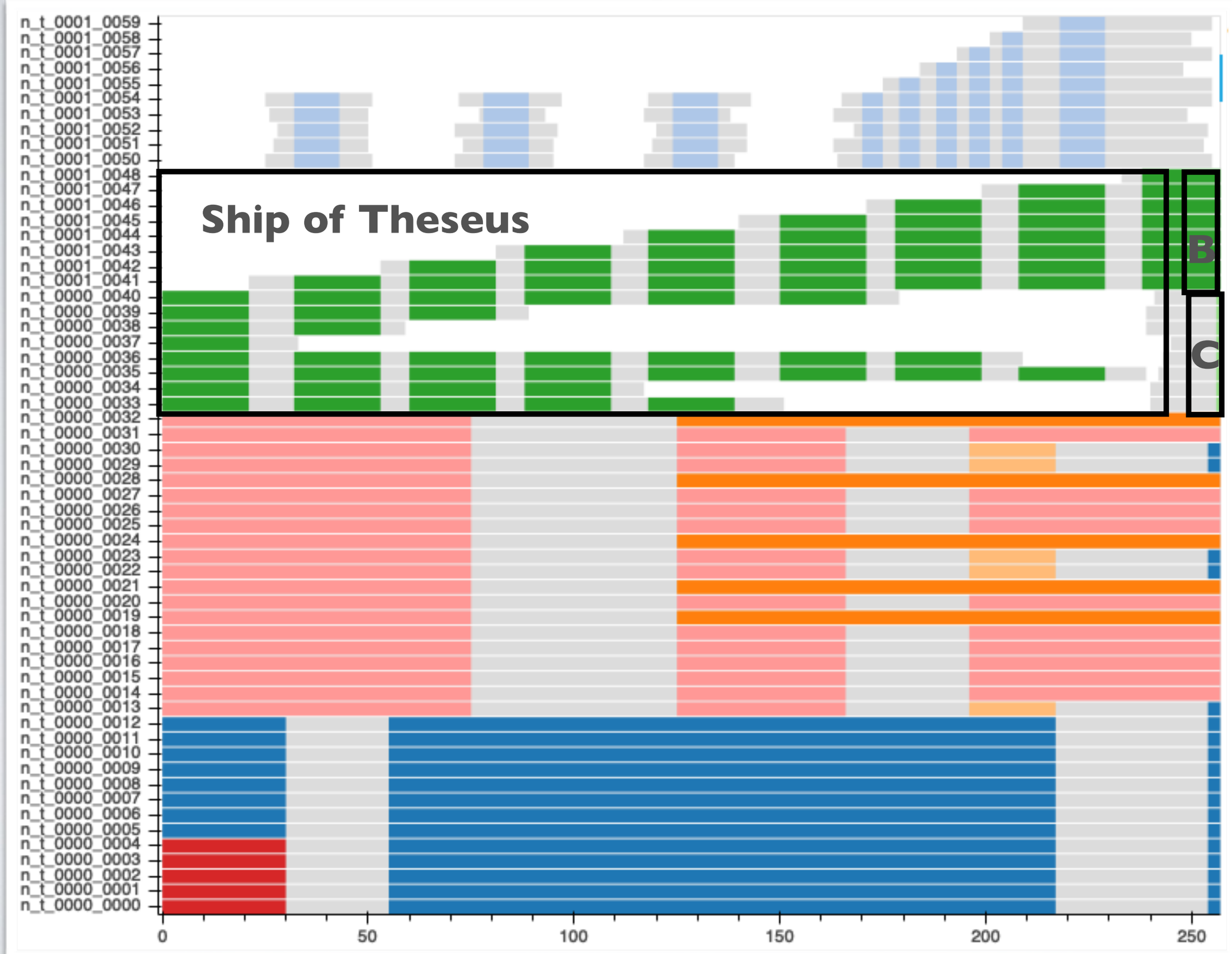
SYNTHETIC NETWORK

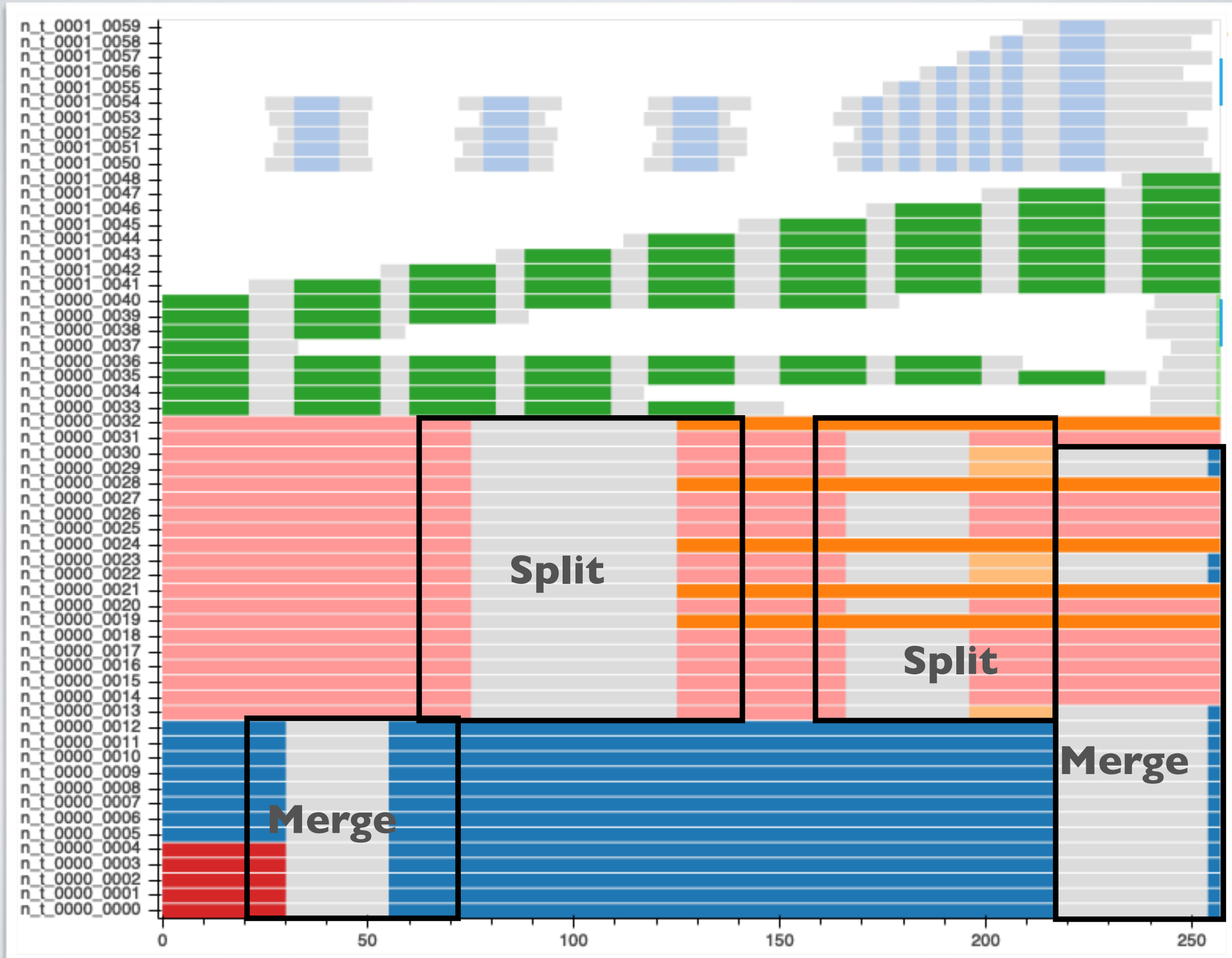
DCD EVALUATION

- Benchmarks allow to generate networks with a controlled community structure
  - Based on LFR benchmark of Stochastic Block Models

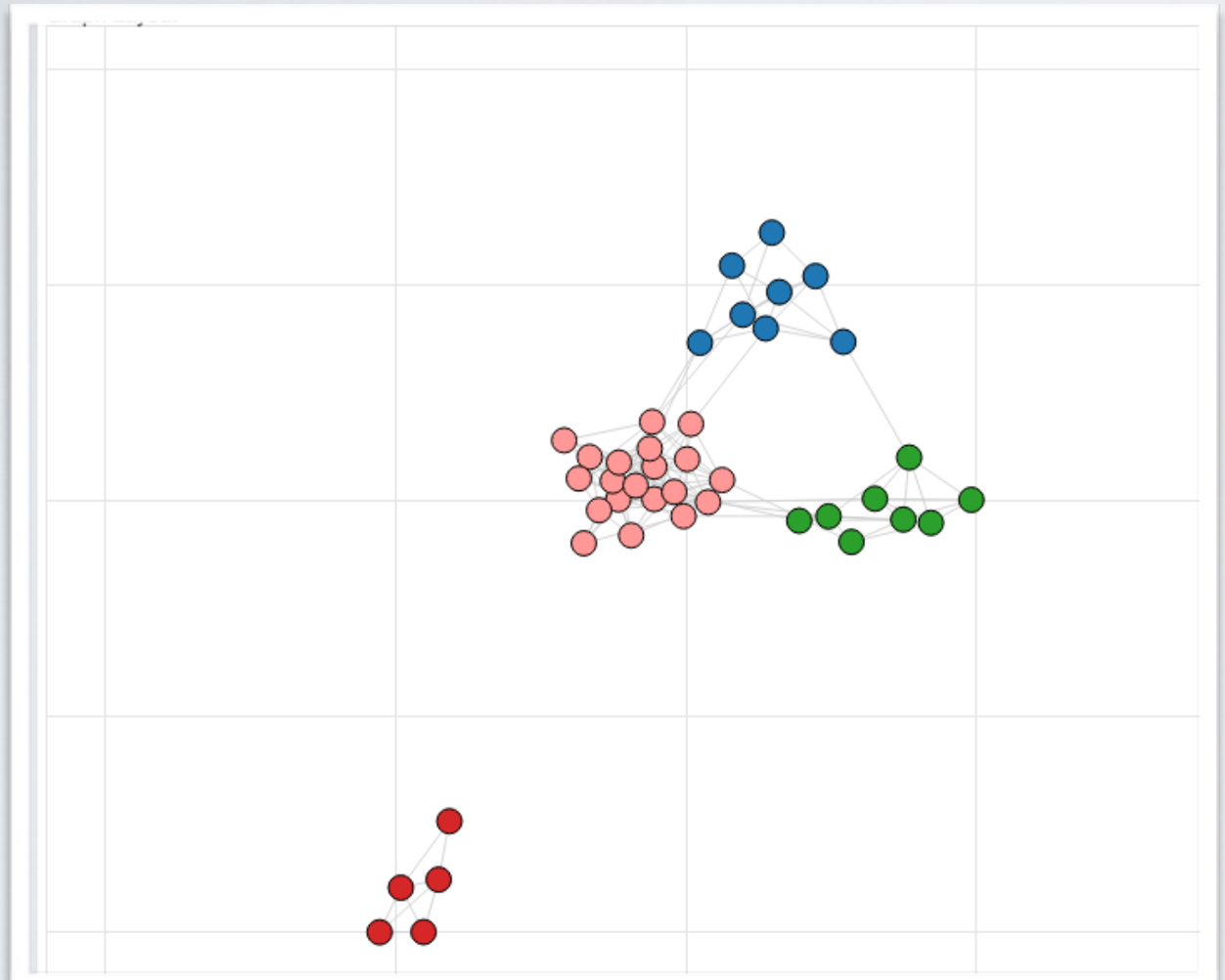

n_t_0001_0059
n_t_0001_0058
n_t_0001_0057
n_t_0001_0056
n_t_0001_0055
n_t_0001_0054
n_t_0001_0053
n_t_0001_0052
n_t_0001_0051
n_t_0001_0050
n_t_0001_0048
n_t_0001_0047
n_t_0001_0046
n_t_0001_0045
n_t_0001_0044
n_t_0001_0043
n_t_0001_0042
n_t_0001_0041
n_t_0000_0040
n_t_0000_0039
n_t_0000_0038
n_t_0000_0037
n_t_0000_0036
n_t_0000_0035
n_t_0000_0034
n_t_0000_0033
n_t_0000_0032
n_t_0000_0031
n_t_0000_0030
n_t_0000_0029
n_t_0000_0028
n_t_0000_0027
n_t_0000_0026
n_t_0000_0025
n_t_0000_0024
n_t_0000_0023
n_t_0000_0022
n_t_0000_0021
n_t_0000_0020
n_t_0000_0019
n_t_0000_0018
n_t_0000_0017
n_t_0000_0016
n_t_0000_0015
n_t_0000_0014
n_t_0000_0013
n_t_0000_0012
n_t_0000_0011
n_t_0000_0010
n_t_0000_0009
n_t_0000_0008
n_t_0000_0007
n_t_0000_0006
n_t_0000_0005
n_t_0000_0004
n_t_0000_0003
n_t_0000_0002
n_t_0000_0001
n_t_0000_0000
0
50
100
150
200
250


Resurgence
Growth
n_t_0001_0059
n_t_0001_0058
n_t_0001_0057
n_t_0001_0056
n_t_0001_0055
n_t_0001_0054
n_t_0001_0053
n_t_0001_0052
n_t_0001_0051
n_t_0001_0050
n_t_0001_0048
n_t_0001_0047
n_t_0001_0046
n_t_0001_0045
n_t_0001_0044
n_t_0001_0043
n_t_0001_0042
n_t_0001_0041
n_t_0000_0040
n_t_0000_0039
n_t_0000_0038
n_t_0000_0037
n_t_0000_0036
n_t_0000_0035
n_t_0000_0034
n_t_0000_0033
n_t_0000_0032
n_t_0000_0031
n_t_0000_0030
n_t_0000_0029
n_t_0000_0028
n_t_0000_0027
n_t_0000_0026
n_t_0000_0025
n_t_0000_0024
n_t_0000_0023
n_t_0000_0022
n_t_0000_0021
n_t_0000_0020
n_t_0000_0019
n_t_0000_0018
n_t_0000_0017
n_t_0000_0016
n_t_0000_0015
n_t_0000_0014
n_t_0000_0013
n_t_0000_0012
n_t_0000_0011
n_t_0000_0010
n_t_0000_0009
n_t_0000_0008
n_t_0000_0007
n_t_0000_0006
n_t_0000_0005
n_t_0000_0004
n_t_0000_0003
n_t_0000_0002
n_t_0000_0001
n_t_0000_0000
0
50
100
150
200
250

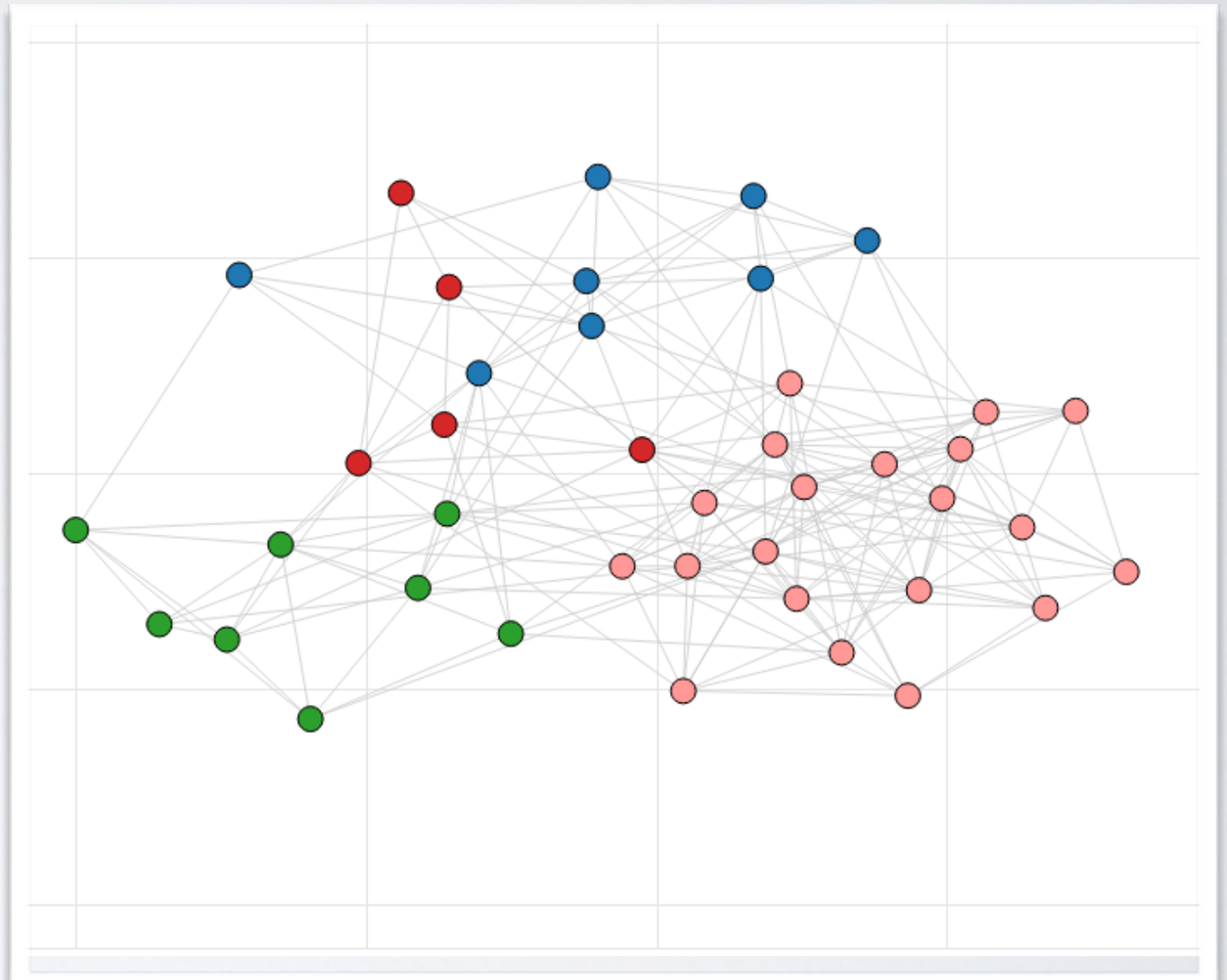

Ship of Theseus
B
C
n_t_0001_0059
n_t_0001_0058
n_t_0001_0057
n_t_0001_0056
n_t_0001_0055
n_t_0001_0054
n_t_0001_0053
n_t_0001_0052
n_t_0001_0051
n_t_0001_0050
n_t_0001_0048
n_t_0001_0047
n_t_0001_0046
n_t_0001_0045
n_t_0001_0044
n_t_0001_0043
n_t_0001_0042
n_t_0001_0041
n_t_0000_0040
n_t_0000_0039
n_t_0000_0038
n_t_0000_0037
n_t_0000_0036
n_t_0000_0035
n_t_0000_0034
n_t_0000_0033
n_t_0000_0032
n_t_0000_0031
n_t_0000_0030
n_t_0000_0029
n_t_0000_0028
n_t_0000_0027
n_t_0000_0026
n_t_0000_0025
n_t_0000_0024
n_t_0000_0023
n_t_0000_0022
n_t_0000_0021
n_t_0000_0020
n_t_0000_0019
n_t_0000_0018
n_t_0000_0017
n_t_0000_0016
n_t_0000_0015
n_t_0000_0014
n_t_0000_0013
n_t_0000_0012
n_t_0000_0011
n_t_0000_0010
n_t_0000_0009
n_t_0000_0008
n_t_0000_0007
n_t_0000_0006
n_t_0000_0005
n_t_0000_0004
n_t_0000_0003
n_t_0000_0002
n_t_0000_0001
n_t_0000_0000
0
50
100
150
200
250


n_t_0001_0059
n_t_0001_0058
n_t_0001_0057
n_t_0001_0056
n_t_0001_0055
n_t_0001_0054
n_t_0001_0053
n_t_0001_0052
n_t_0001_0051
n_t_0001_0050
n_t_0001_0048
n_t_0001_0047
n_t_0001_0046
n_t_0001_0045
n_t_0001_0044
n_t_0001_0043
n_t_0001_0042
n_t_0001_0041
n_t_0000_0040
n_t_0000_0039
n_t_0000_0038
n_t_0000_0037
n_t_0000_0036
n_t_0000_0035
n_t_0000_0034
n_t_0000_0033
n_t_0000_0032
n_t_0000_0031
n_t_0000_0030
n_t_0000_0029
n_t_0000_0028
n_t_0000_0027
n_t_0000_0026
n_t_0000_0025
n_t_0000_0024
n_t_0000_0023
n_t_0000_0022
n_t_0000_0021
n_t_0000_0020
n_t_0000_0019
n_t_0000_0018
n_t_0000_0017
n_t_0000_0016
n_t_0000_0015
n_t_0000_0014
n_t_0000_0013
n_t_0000_0012
n_t_0000_0011
n_t_0000_0010
n_t_0000_0009
n_t_0000_0008
n_t_0000_0007
n_t_0000_0006
n_t_0000_0005
n_t_0000_0004
n_t_0000_0003
n_t_0000_0002
n_t_0000_0001
n_t_0000_0000
0
50
100
150
200
250
Split
Split
Merge
Merge

BENCHMARK



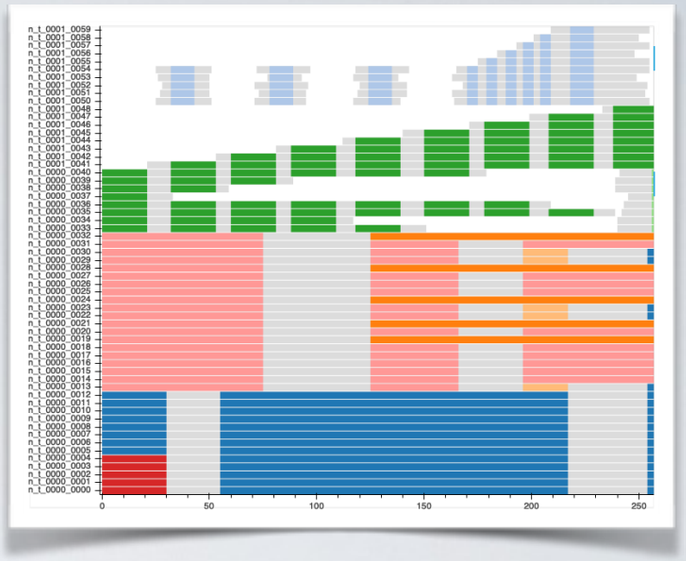
(b) The static graph at time t=0, version *sharp* ($\alpha = 0.9, \mu = 0.05$)

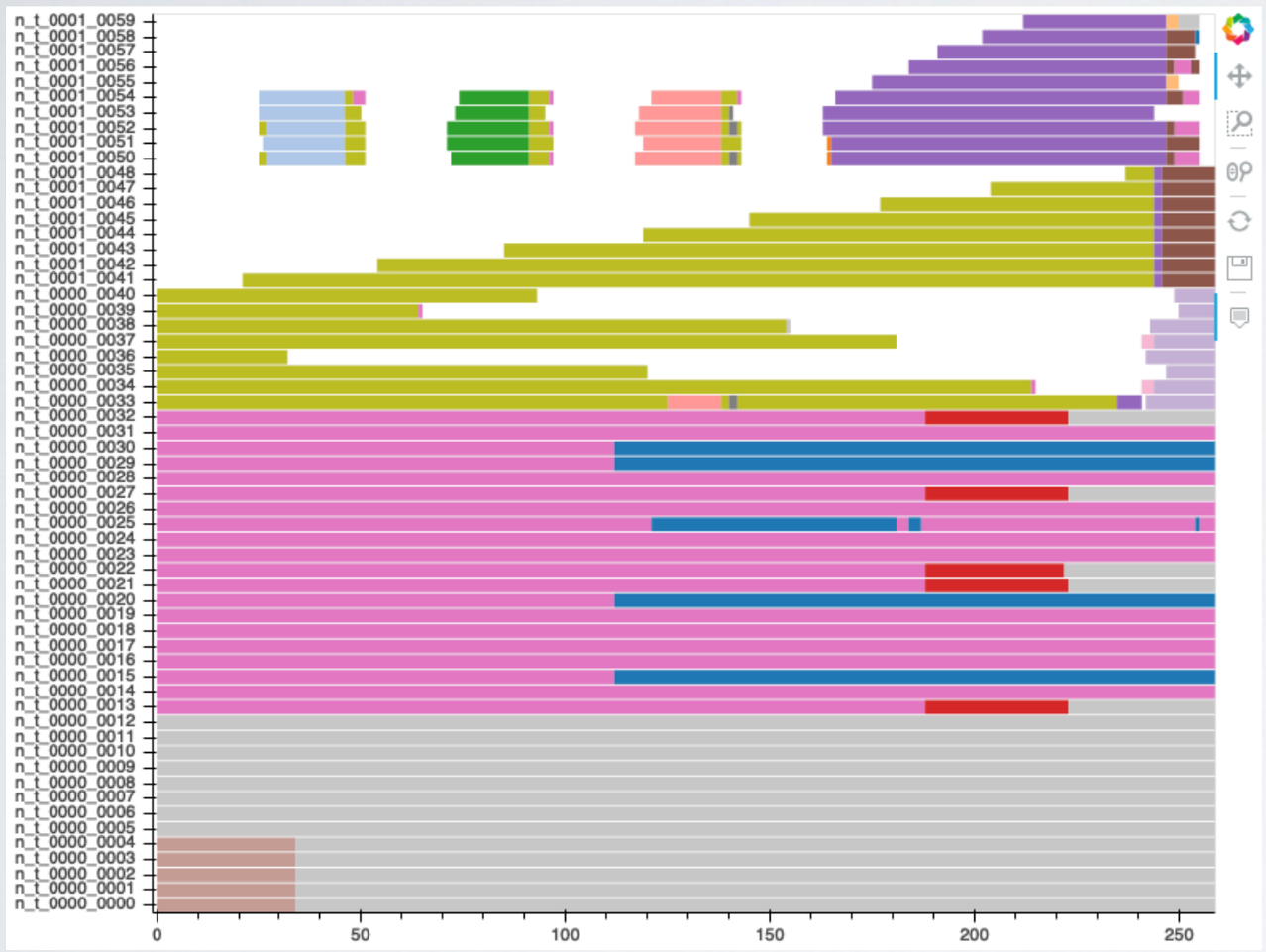


(c) The static graph at time t=0, version *blurred* ($\alpha = 0.8, \mu = 0.25$)

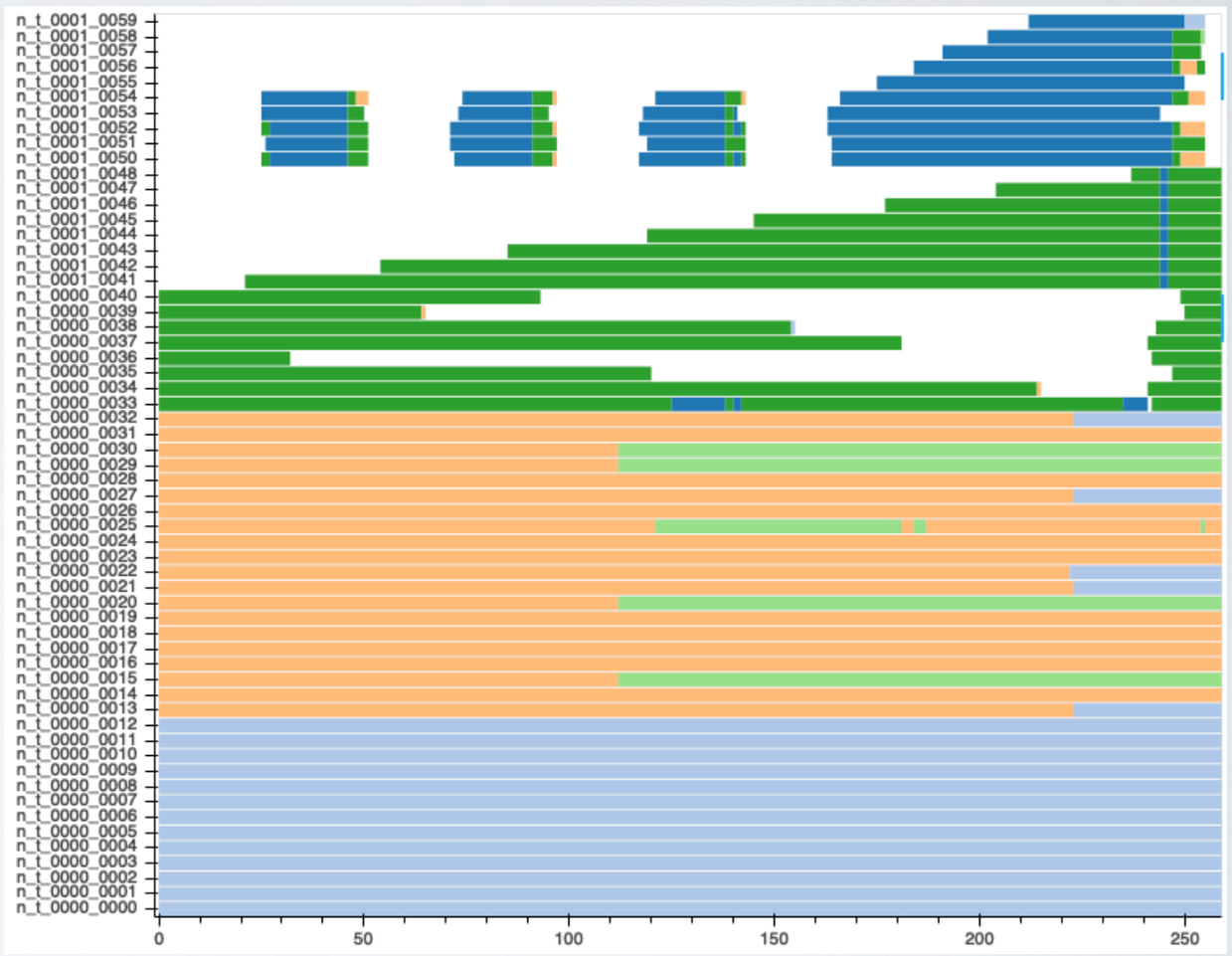
RESULTS WITH SHARP COMMUNITIES

SHARP



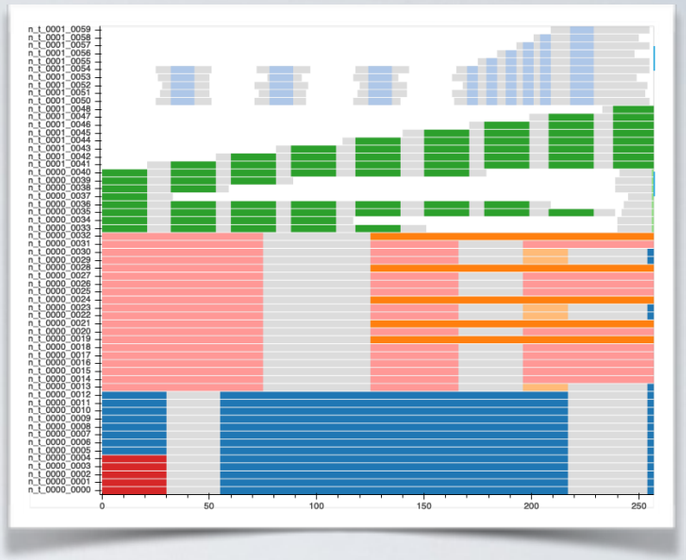


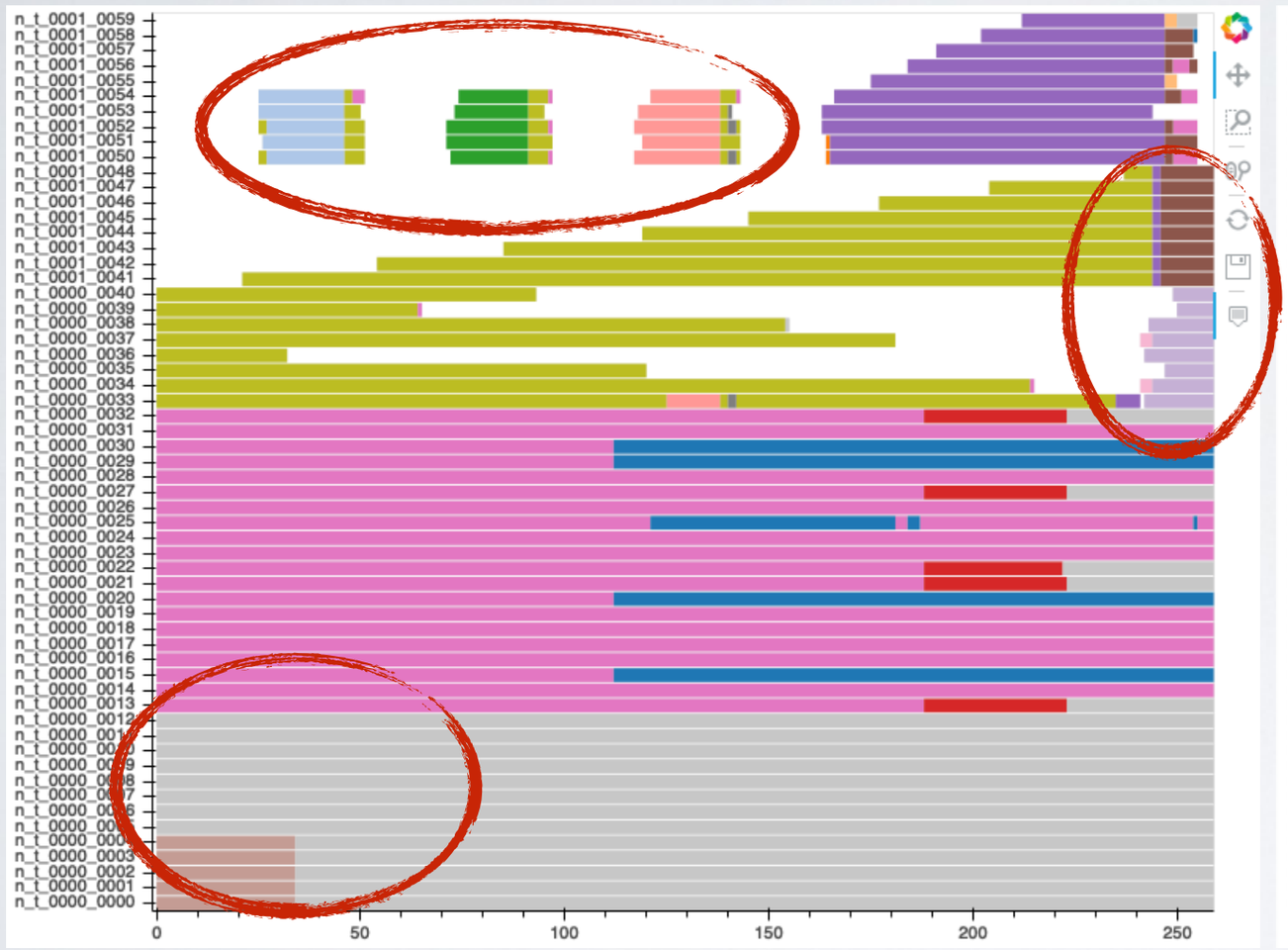
(a) No-smoothing



(b) Label Smoothing

SHARP





(a) No-smoothing

(b) Label Smoothing

RESULTS WITH BLURRED COMMUNITIES

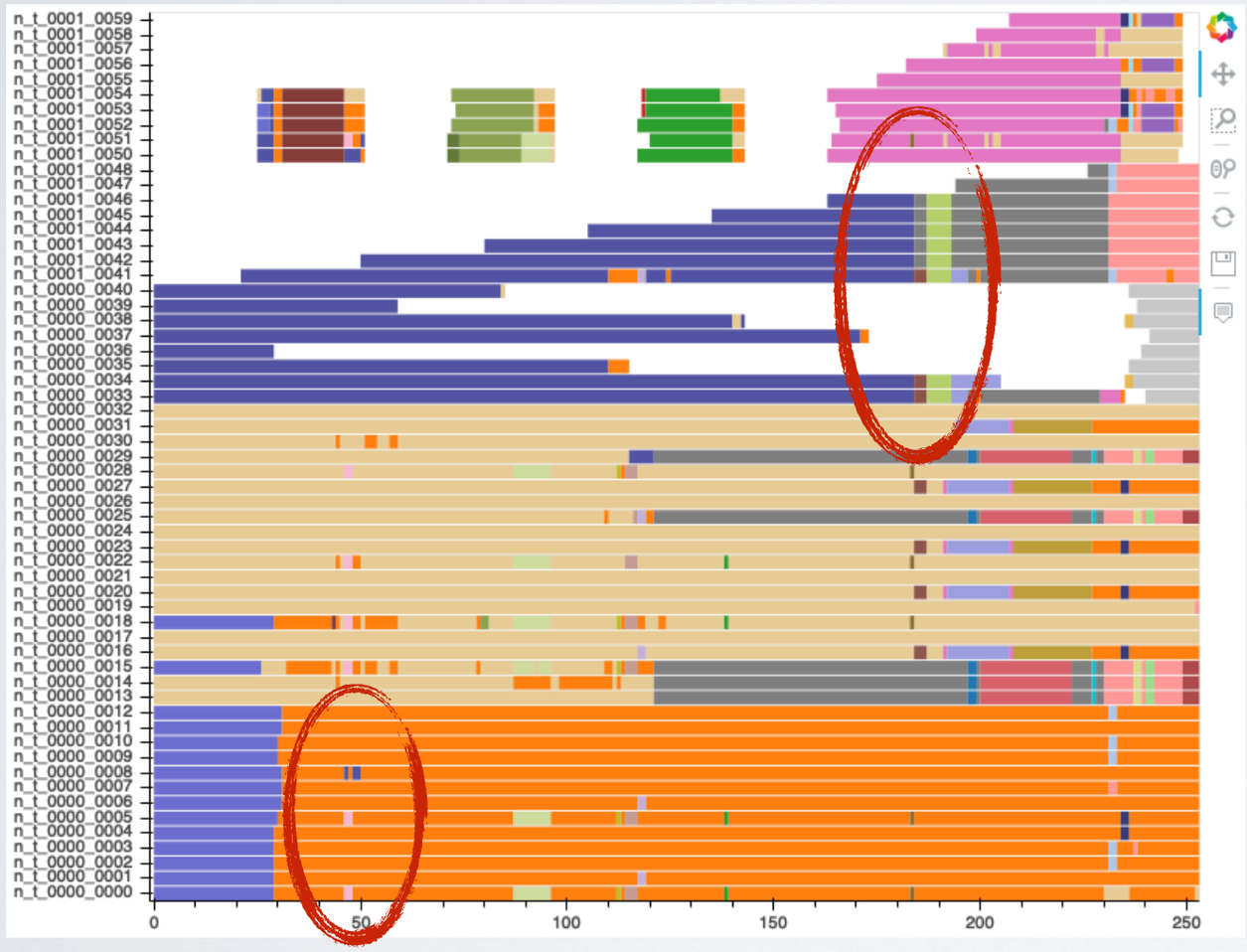
BLURRED

(a) No-smoothing

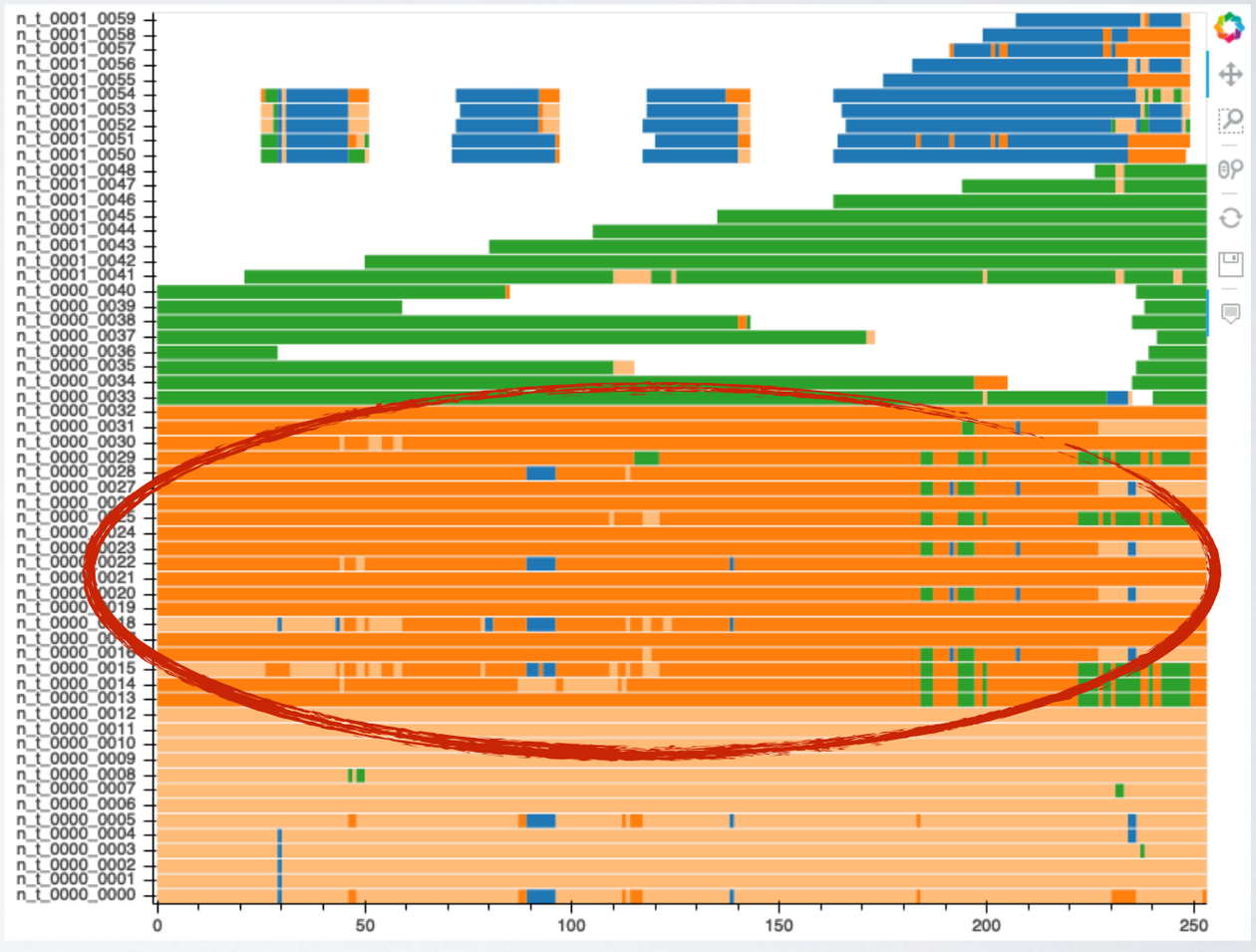
(b) Label Smoothing

BLURRED

Identity loss



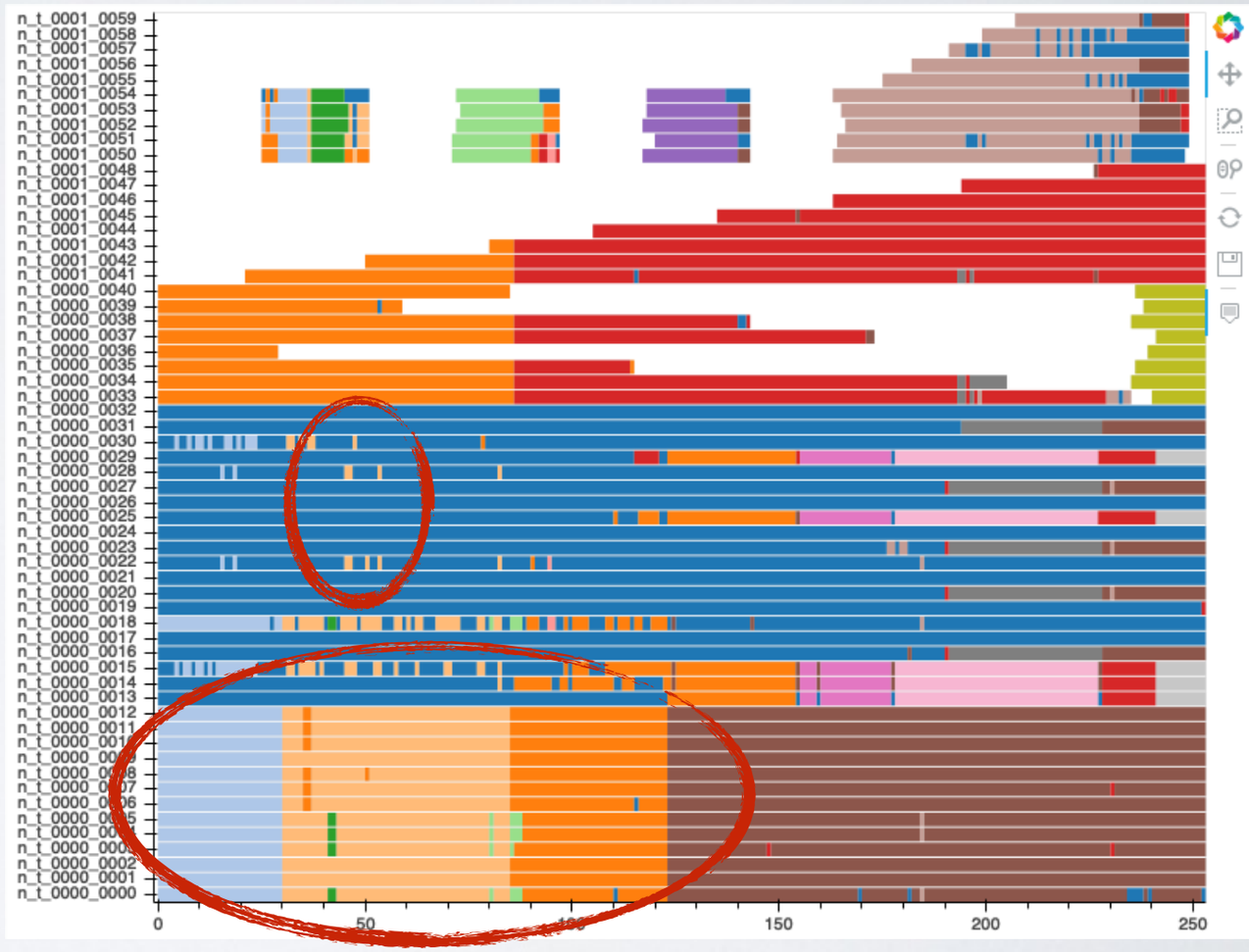
(a) No-smoothing



(b) Label Smoothing

Glitches

(c) DYNAMO



(d) Transversal Network

(e) Implicit Global

(f) DYNMOGA

TO SUM UP ON DYNAMIC GRAPHS

TO SUM UP

- Currently, most practitioners still use the snapshot approaches
- Most researchers agree that it has many drawbacks
- But currently:
  - No single framework
  - No library
  - Dataset not as ubiquitous as static graphs
    - (often privates…)