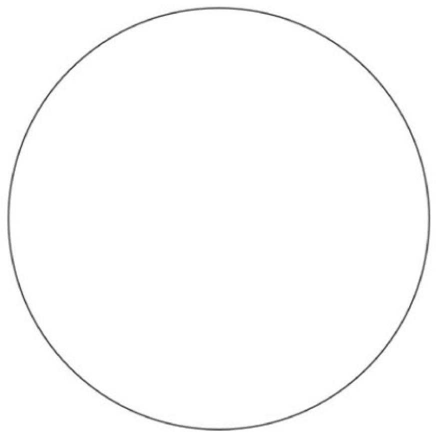
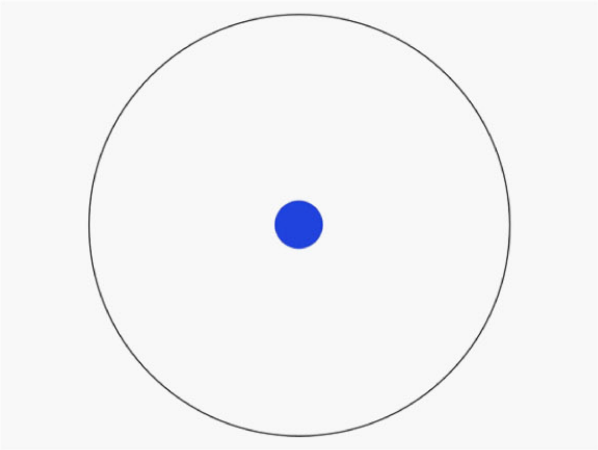
A RESEARCH QUESTION

Communities in degenerate link streams

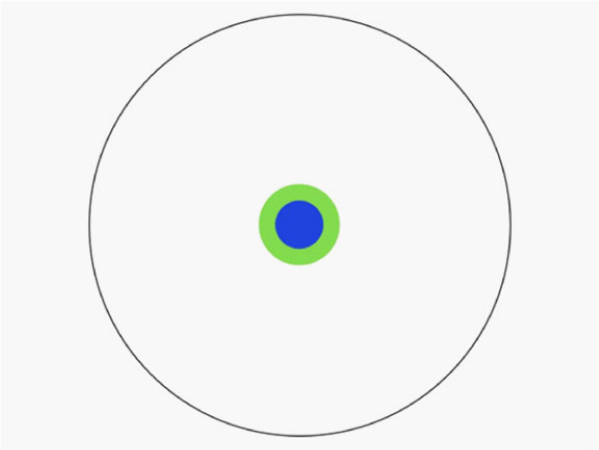
Imagine a circle that contains all of human knowledge:



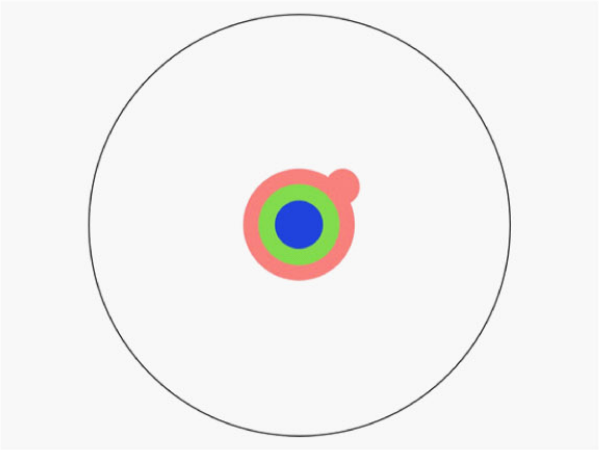
By the time you finish elementary school, you know a little:



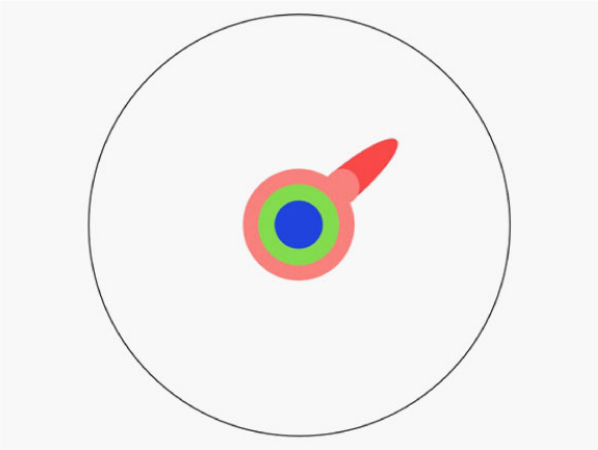
By the time you finish high school, you know a bit more:



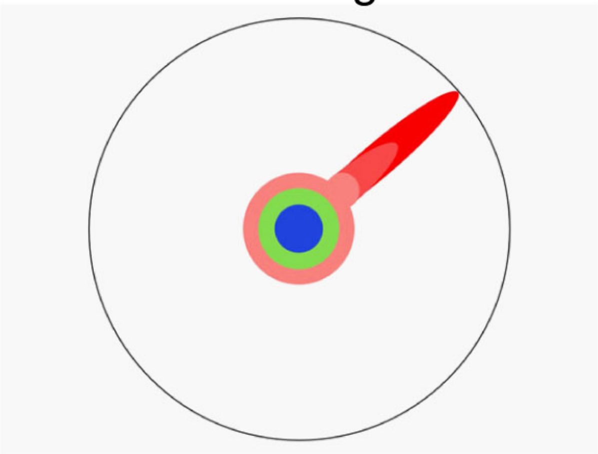
With a bachelor's degree, you gain a specialty:



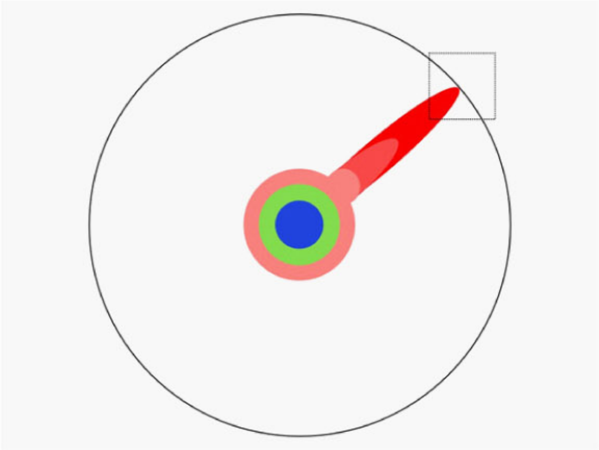
A master's degree deepens that specialty:



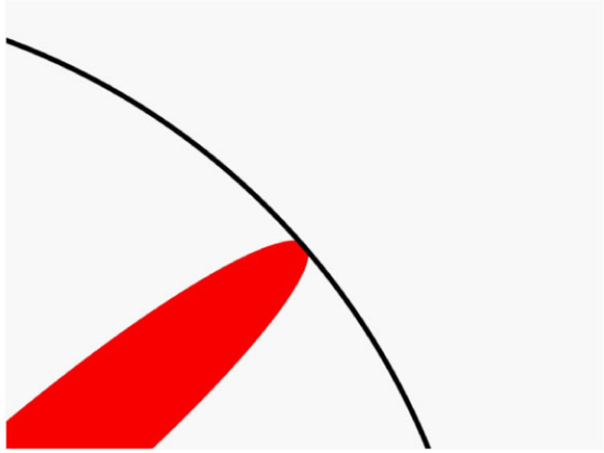
Reading research papers takes you to the edge of human knowledge:



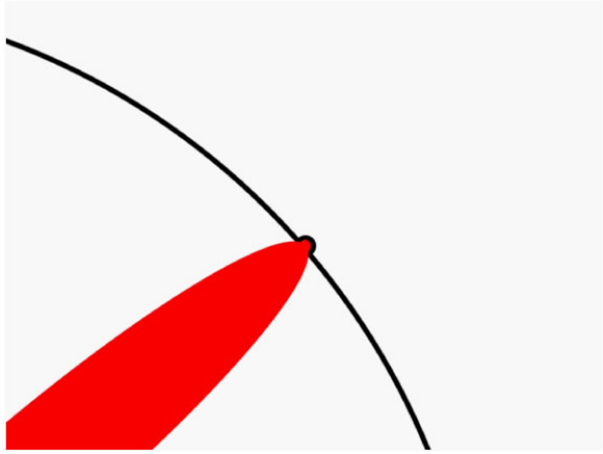
Once you're at the boundary, you focus:



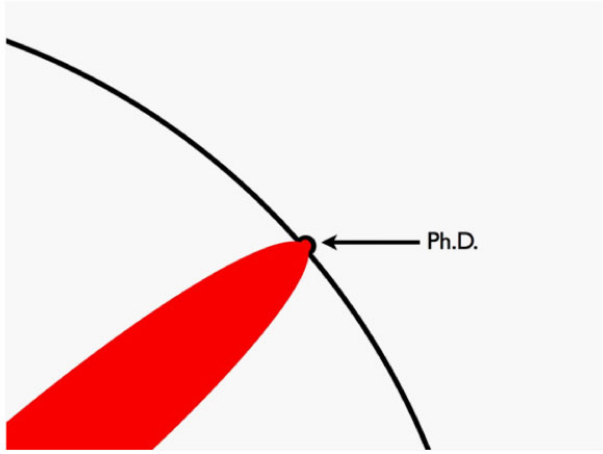
You push at the boundary for a few years:



Until one day, the boundary gives way:

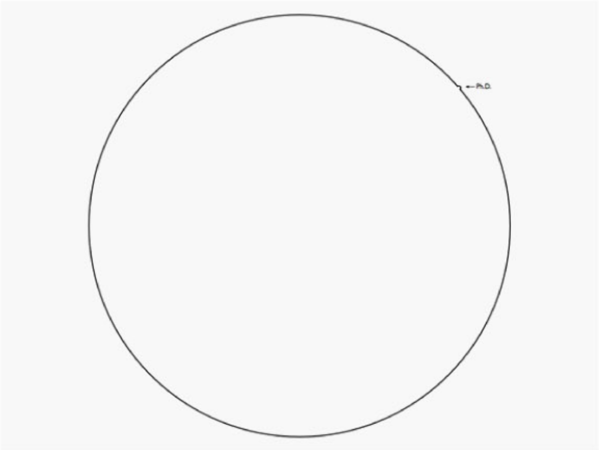


And, that dent you've made is called a Ph.D.:



Of course, the world looks different to you now:

So, don't forget the bigger picture:



Keep pushing.

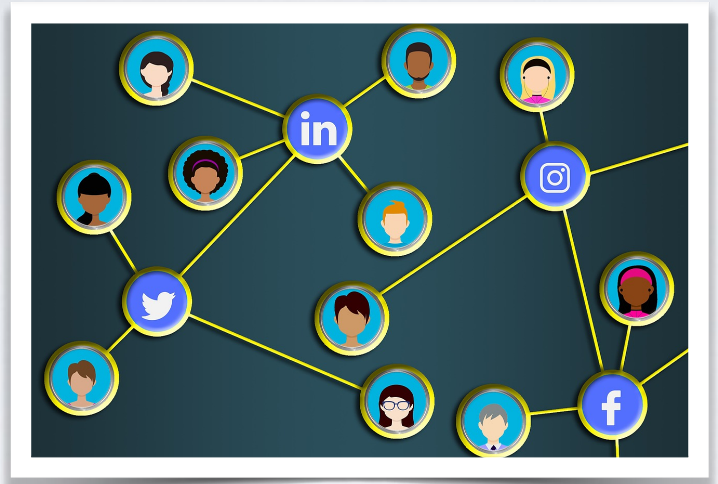
DYNAMIC NETWORKS

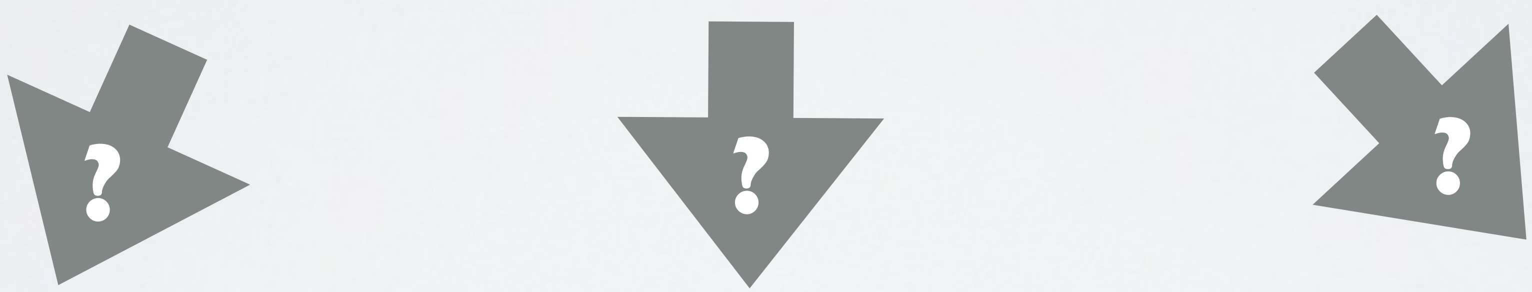
- Most real world networks are dynamic
  - Facebook friendship
    - People joining/leaving
    - Friend/Unfriend
  - Twitter mention network
    - Each mention has a timestamp
    - Aggregated every day/month/year => still dynamic
  - World Wide Web
  - Urban network
  - …

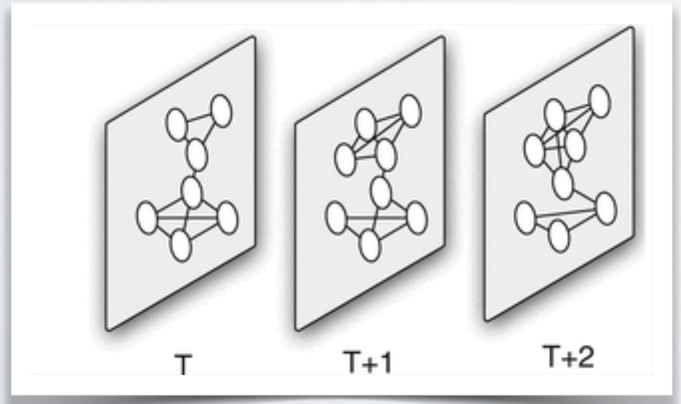
DYNAMIC NETWORKS

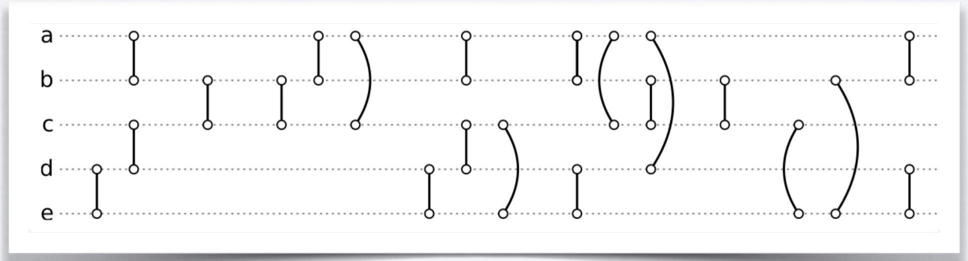
- Most real world networks are dynamic
  - Nodes can appear/disappear
  - Edges can appear/disappear
  - Nature of relations can change
- How to represent those changes?
- How to manipulate dynamic networks?

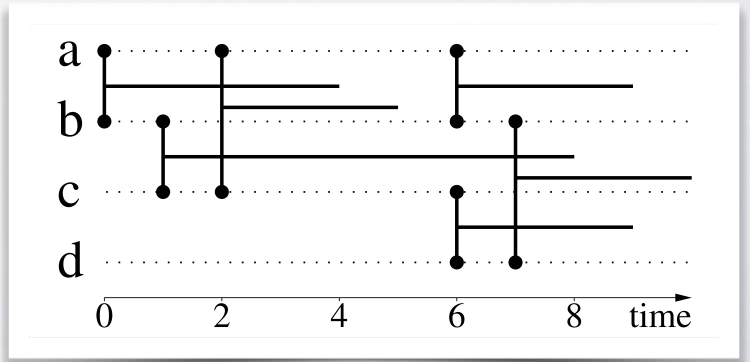
SEVERAL FORMALISMS











TEMPORAL NETWORK

Collected dataset, for instance in (t,u,v) format

Time	u	v
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

Examples:
-SocioPatterns
-Enron
-…

TEMPORAL NETWORK

Snapshots

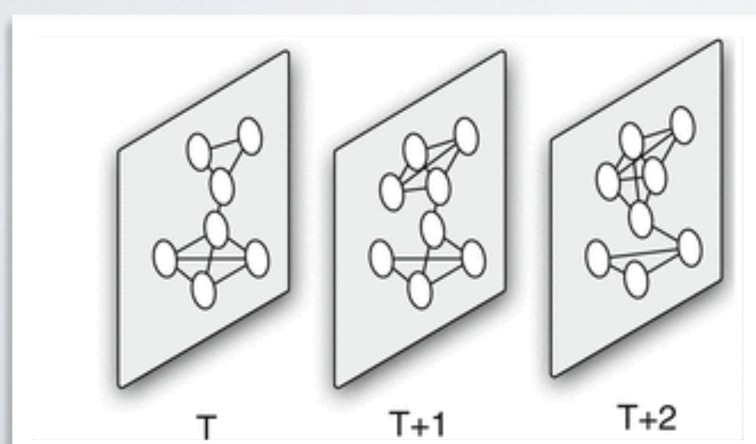
1353304100	1148	1644
1353304100	1613	1672
1353304100	656	682
1353304100	1632	1671
1353304120	1492	1613
1353304120	656	682
1353304120	1632	1671
1353304140	1148	1644
1353304160	656	682
1353304160	1108	1601
1353304160	1632	1671
1353304160	626	698

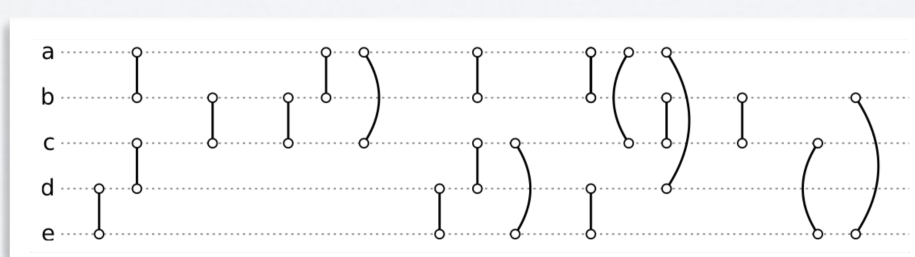
Link Stream

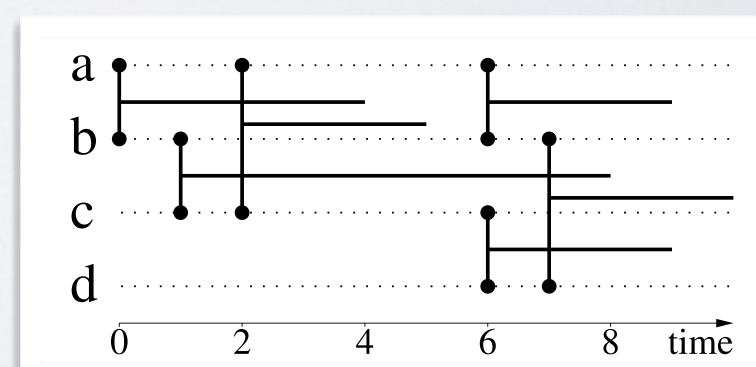
1353304100	1148	1644
1353304100	1613	1672
1353304100	656	682
1353304100	1632	1671
1353304120	1492	1613
1353304120	656	682
1353304120	1632	1671
1353304140	1148	1644
1353304160	656	682
1353304160	1108	1601
1353304160	1632	1671
1353304160	626	698

Interval Graph

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







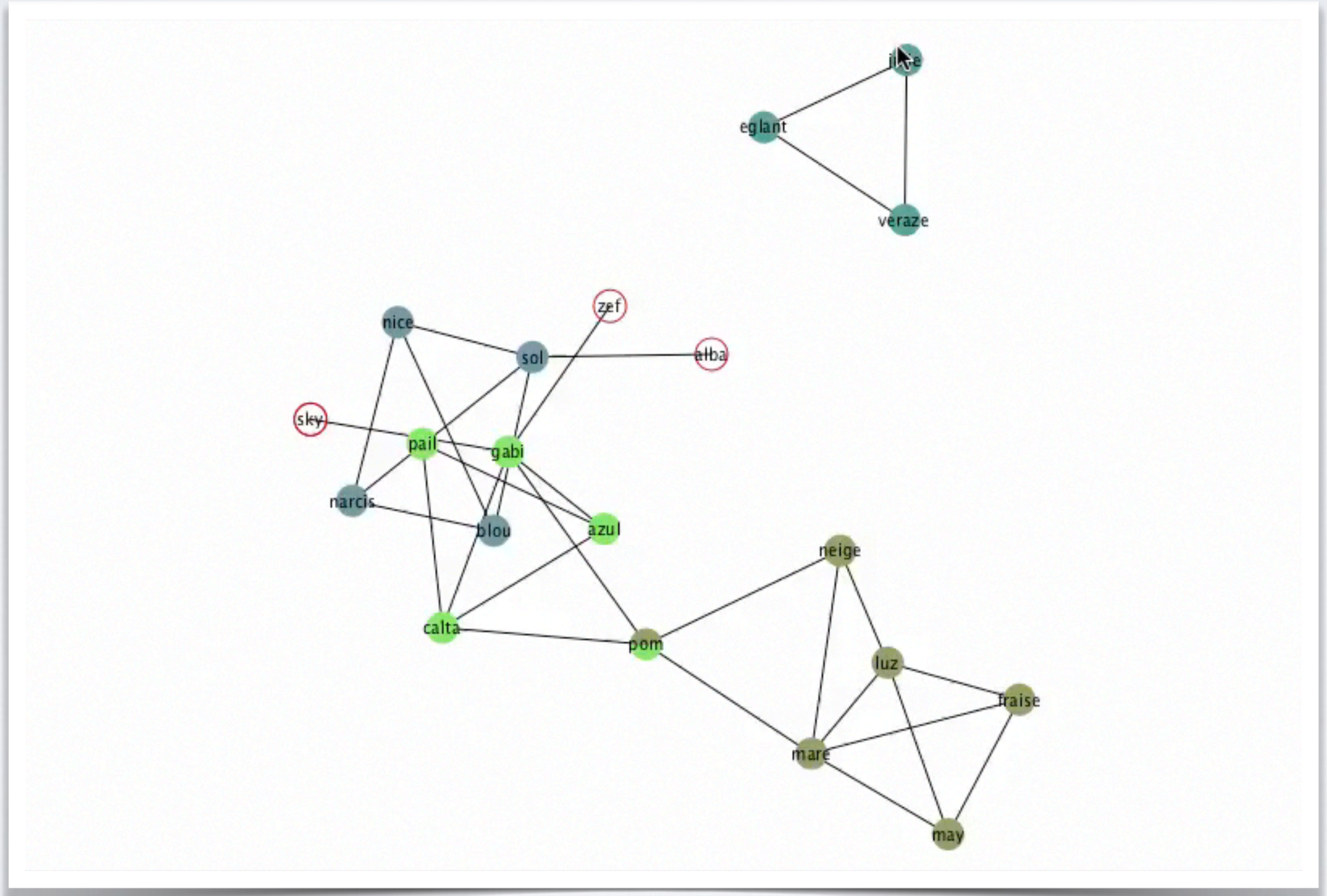
SLOWLY EVOLVING NETWORKS (SEN)

SLOWLY EVOLVING NETWORKS

- Edges change (relatively) slowly
- The network is well defined at any t
  - 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)

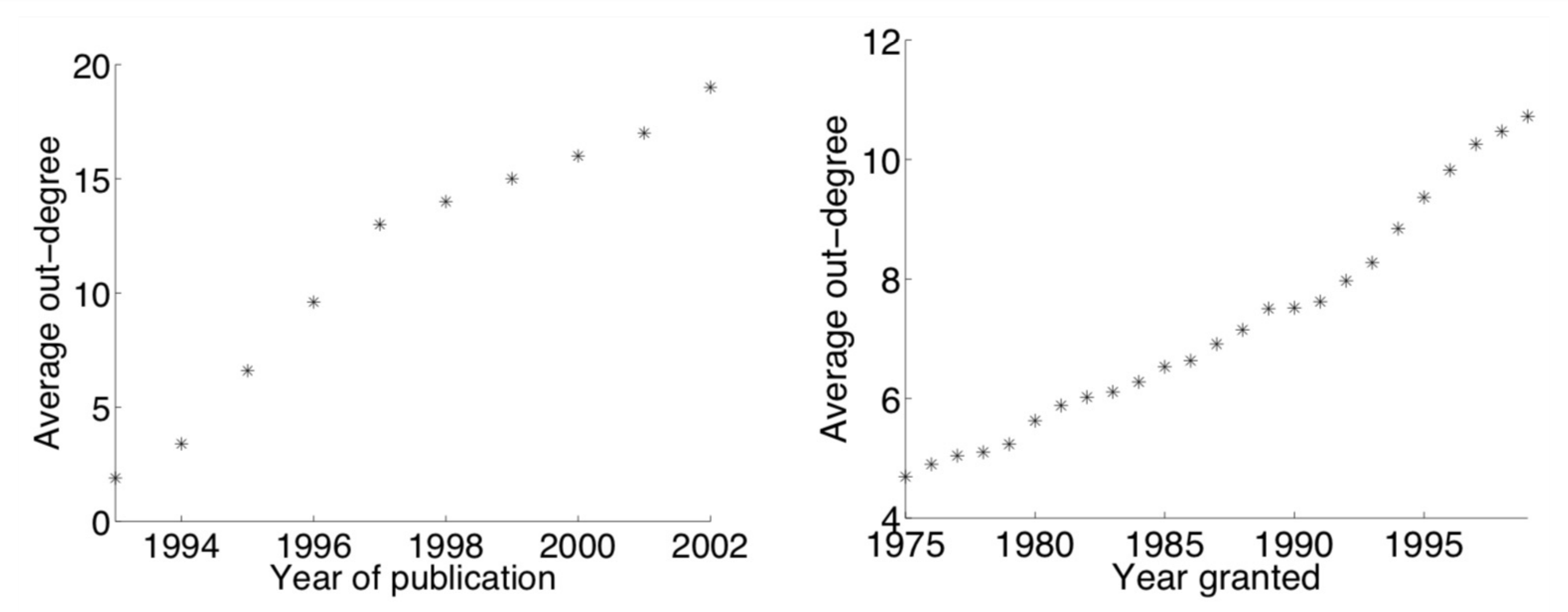
SLOWLY EVOLVING NETWORKS

- Visualization
  - Problem of stability of node positions

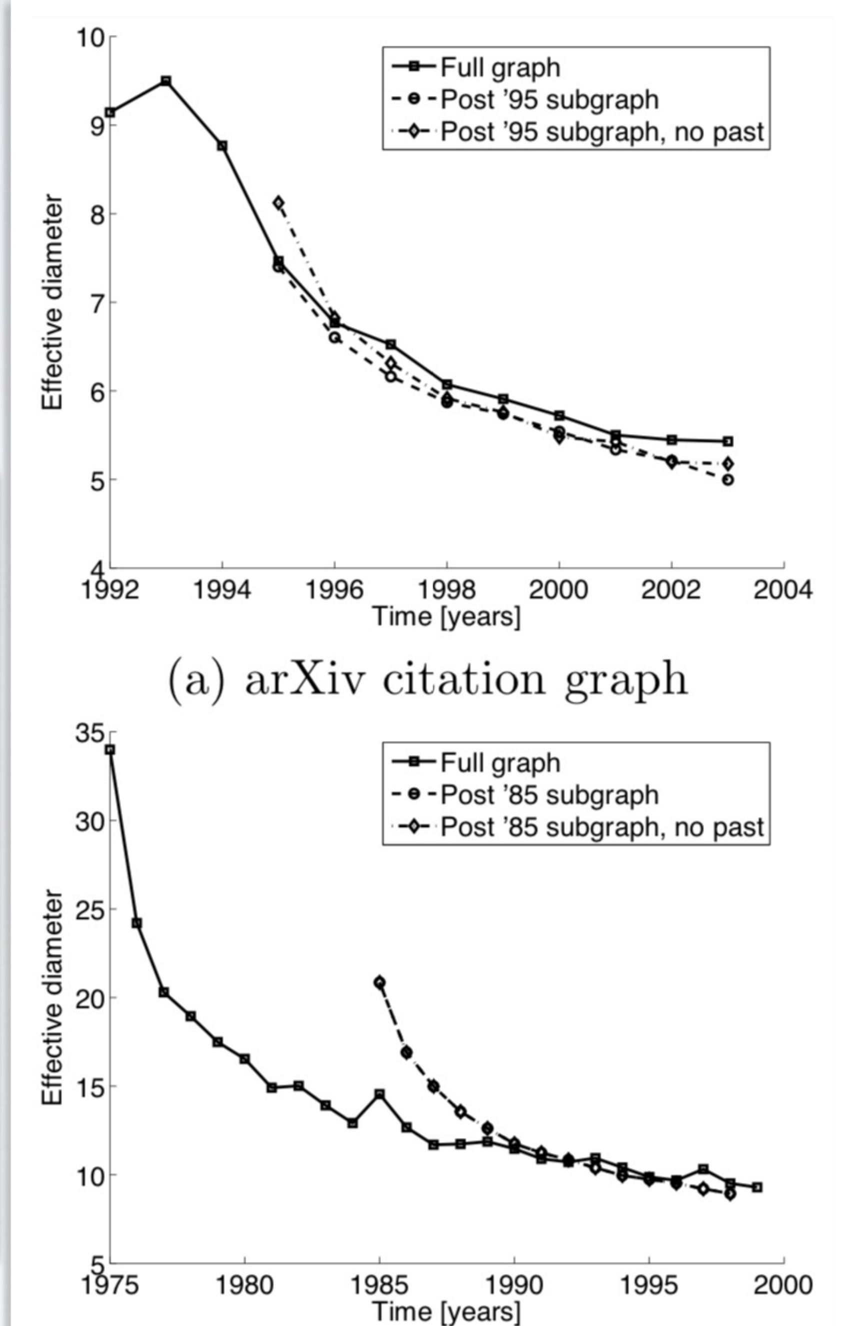


SLOWLY EVOLVING NETWORKS

- Global graph properties



(a) arXiv (b) Patents



(a) arXiv citation graph

(c) Patents citation graph

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.

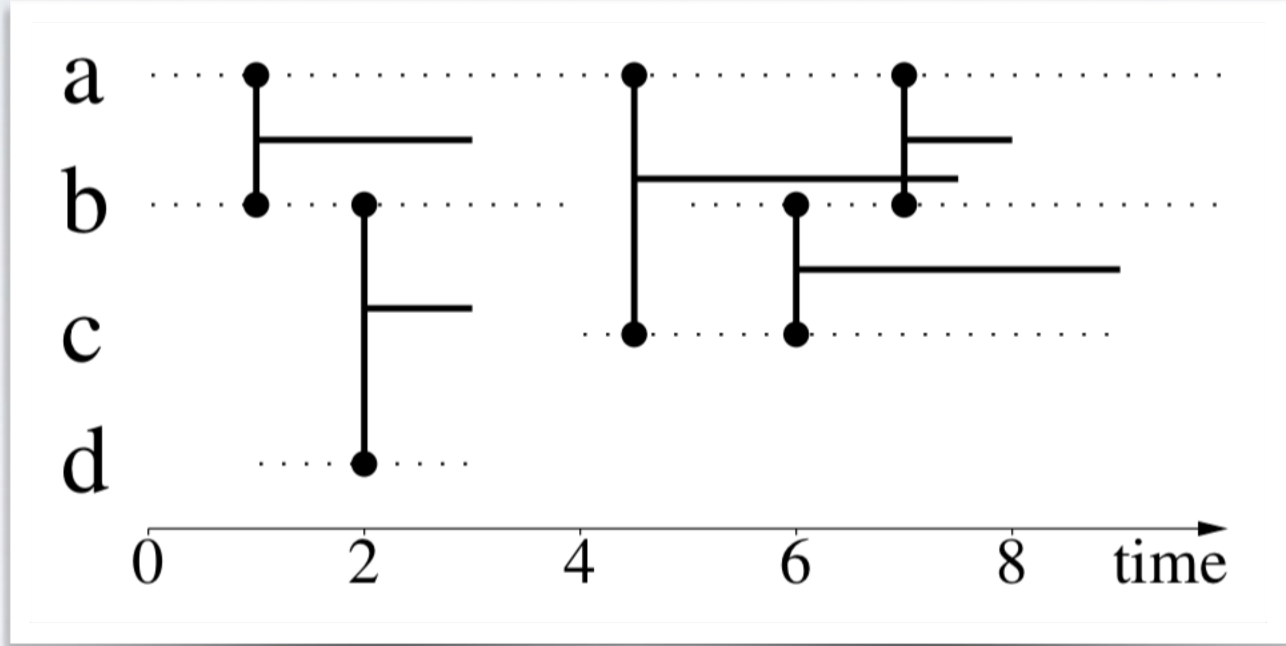
UNSTABLE/DEGENERATE TEMPORAL NETWORKS

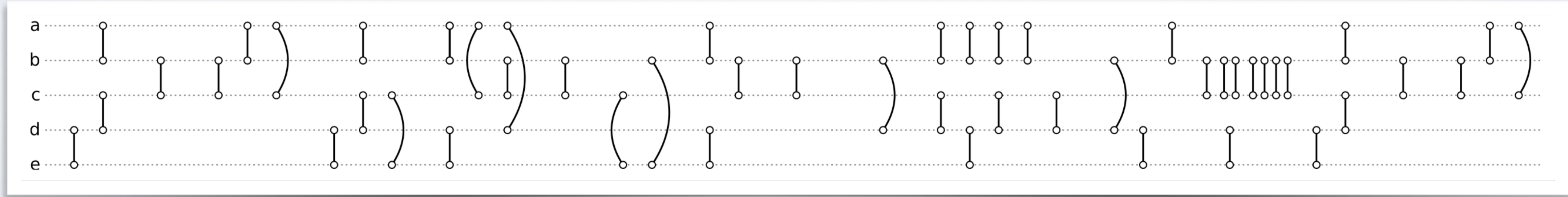
Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. "Stream graphs and link streams for the modeling of interactions over time". In: *Social Network Analysis and Mining* 8.1 (2018), p. 61.

UNSTABLE TEMPORAL NETWORK

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

UNSTABLE TEMPORAL NETWORK





UNSTABLE TEMPORAL NETWORK

- Common solution: transform into SEN using aggregation/ sliding windows
  - Information loss
  - How to chose a proper aggregation window size?
- New theoretical tools developed to deal with such networks

> - **Link Streams** & **Stream Graphs** (Latapy, Viard, and Magnien 2018)
> - **Temporal Networks**, **Contact Sequences** and **Interval Graphs** (Holme and Saramäki 2012)
> - **Time Varying Graphs** (Casteigts et al. 2012)

CENTRALITIES
&
NETWORK PROPERTIES
IN STREAM GRAPHS

STREAM GRAPHS

Stream Graph (SG)- Definition

Stream Graphs have been proposed in[a] as a generic formalism – it can represent any type of dynamic networks, continuous, discrete, with or without duration, with the objective or redefining typical notions of graphs on dynamic networks, including degenerate ones.
Let's define a Stream Graph

$$S = (T, V, W, E)$$

T	**Set of Possible times** (Discrete or Time intervals)
V	**Set of Nodes**
W	**Vertices presence time** $V \times T$
E	**Edges presence time** $V \times V \times T$

[a] Latapy, Viard, and Magnien 2018.

STREAM GRAPHS

SG - Time-Entity designation

It is useful to work with Stream Graphs to introduce some new notions mixing entities (nodes, edges) and time:

V_t	**Nodes At Time**: set of nodes present at time t
E_t	**Edges At Time**: set of edges present at time t
G_t	**Snapshot**: Graph at time t, $G_t = (V_t, E_t)$
v_t	**Node-time**: v_t exists if node v is present at time t
$(u, v)_t$	**Edge-time**: $(u, v)_t$ exists if edge (u, v) is present at time t
T_u	**Times Of Node**: the set of times during which u is present
T_{uv}	**Times Of Edge**: the set of times during which edge (u, v) is present

STREAM GRAPHS

N_u	**Node presence**: The fraction of the total time during which u is present in the network $\frac{\lvert T_u \rvert}{\lvert T \rvert}$
L_{uv}	**Edge presence**: The fraction of the total time during which (u, v) is present in the network $\frac{\lvert T_{uv} \rvert}{\lvert T \rvert}$

STREAM GRAPHS

SG - Redefining Graph notions

The general idea of redefining static network properties on Stream Graphs is that if the network stays unchanged along time, then properties computed on the stream graph should yield the same values as the same property computed on the aggregated graph.

STREAM GRAPHS

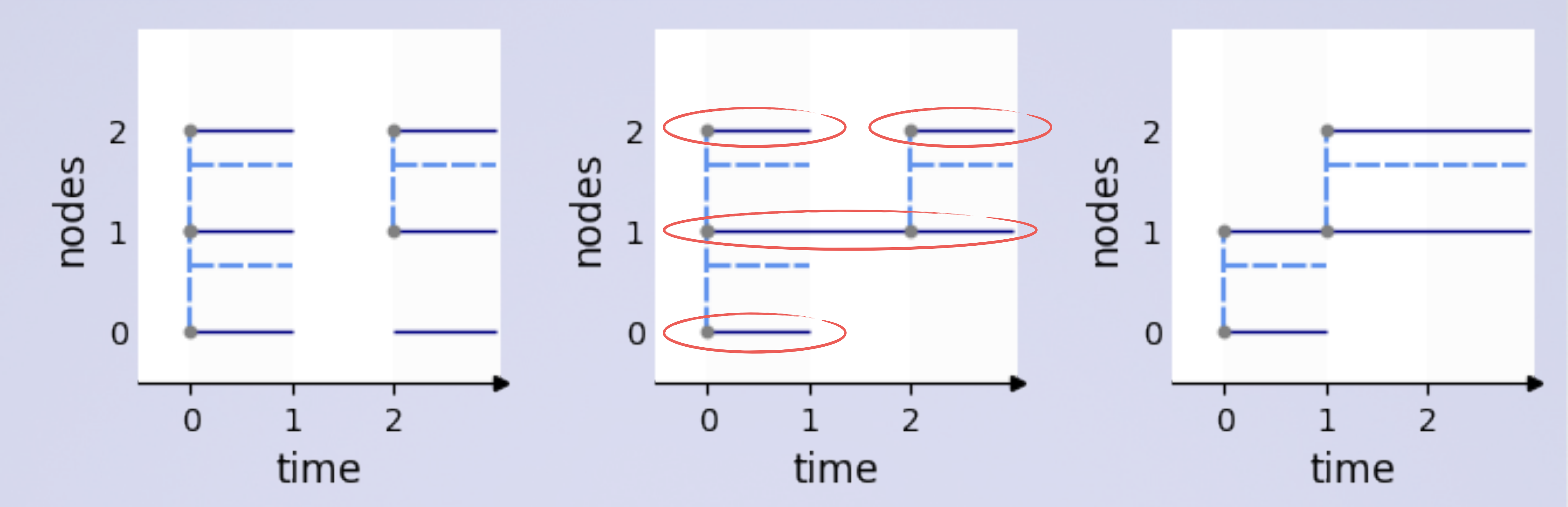
SG - N & L

The number/quantity of nodes in a stream graph is defined as the total presence time of nodes divided by the dataset duration. In general, it isn't an integer.
More formally:

$$N = \sum_{v \in V} N_v = \frac{|W|}{|T|}$$

For instance, $N = 2$ if there are 4 nodes present half the time, or two nodes present all the time.

STREAM GRAPHS



$$N = 2$$

STREAM GRAPHS

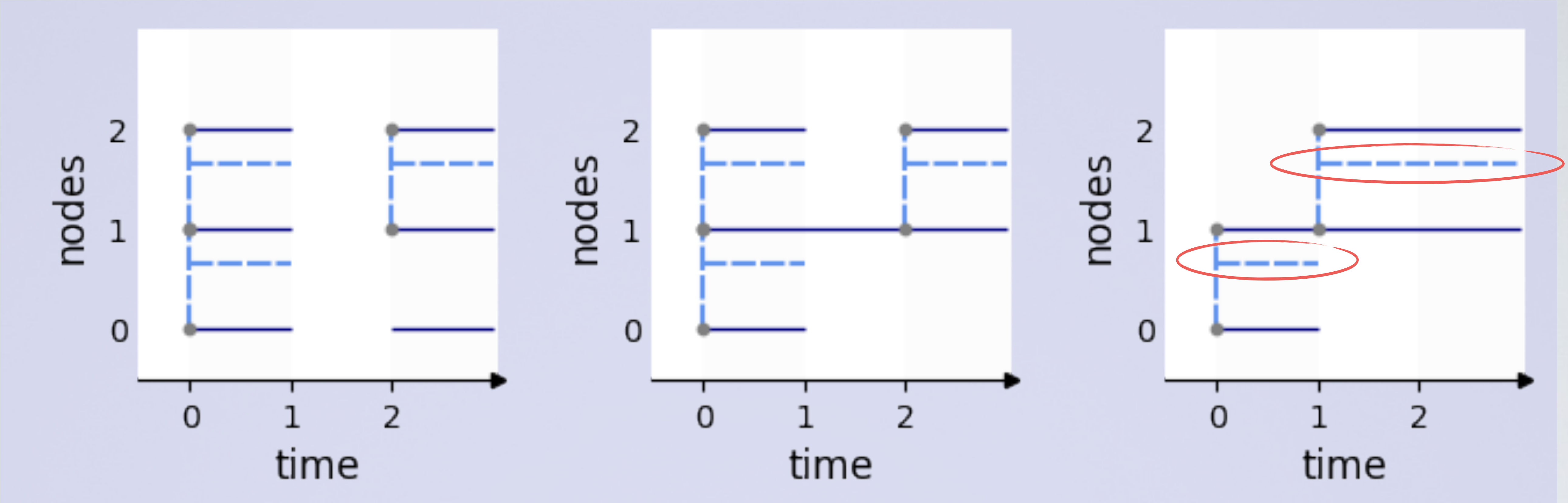
SG - L

The number of edges is defined as the total presence of edges divided by the total dataset duration.
More formally:

$$L = \sum_{(u,v),u,v \in V} L_{uv} = \frac{|E|}{|T|}$$

For instance, $L = 2$ if there are 4 edges present half the time, or two edges present all the time.

STREAM GRAPHS



$$L = 1$$

STREAM GRAPHS

SG - Edge domain - $L_{\max}$

In Stream Graphs, several possible definitions of $L_{\max}$ could exist:

- Ignoring nodes duration: $L_{\max}^1 = |V|^2$
- Ignoring nodes co-presence $L_{\max}^2 = N^2$
- Taking nodes co-presence into account: $L_{\max}{}^3 = \sum_{(u,v),u,v\in V} |T_u \bigcap T_v|$

STREAM GRAPHS

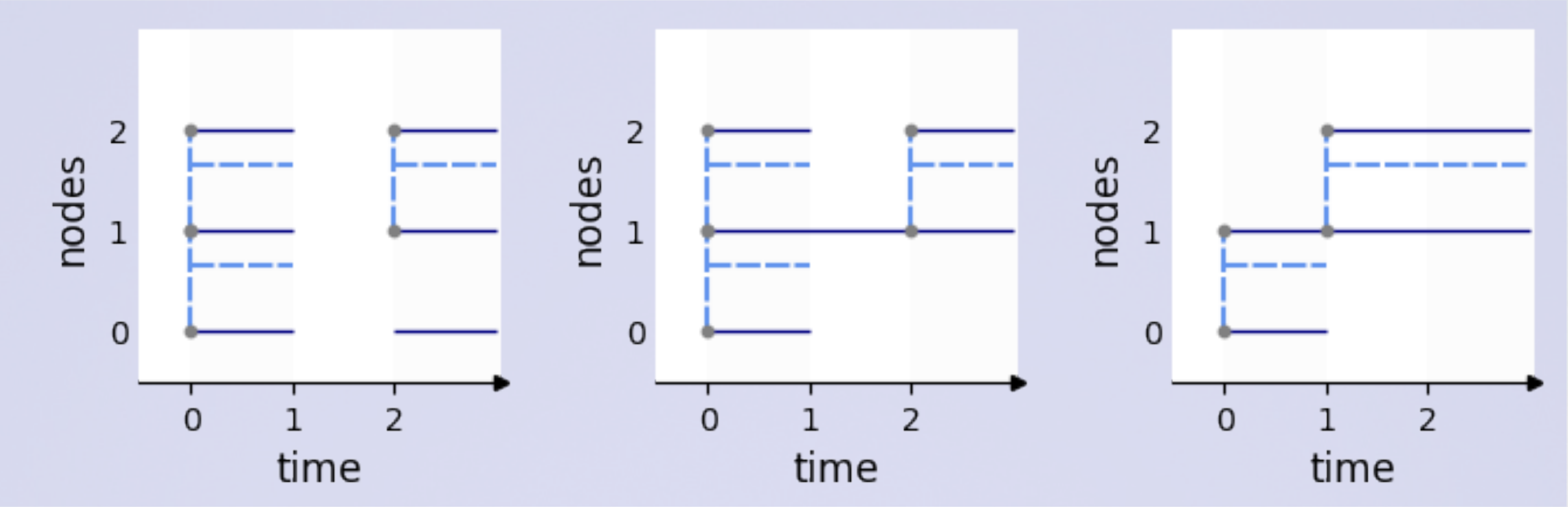
The density in static networks can be understood as the fraction of existing edges among all possible edges,

$$d = \frac{L}{L_{\max}}.$$

In the following, we will use L^3_{max}, as in Latapy et al.

STREAM GRAPHS

$N = 2$ $\qquad\qquad$ $L = 1$

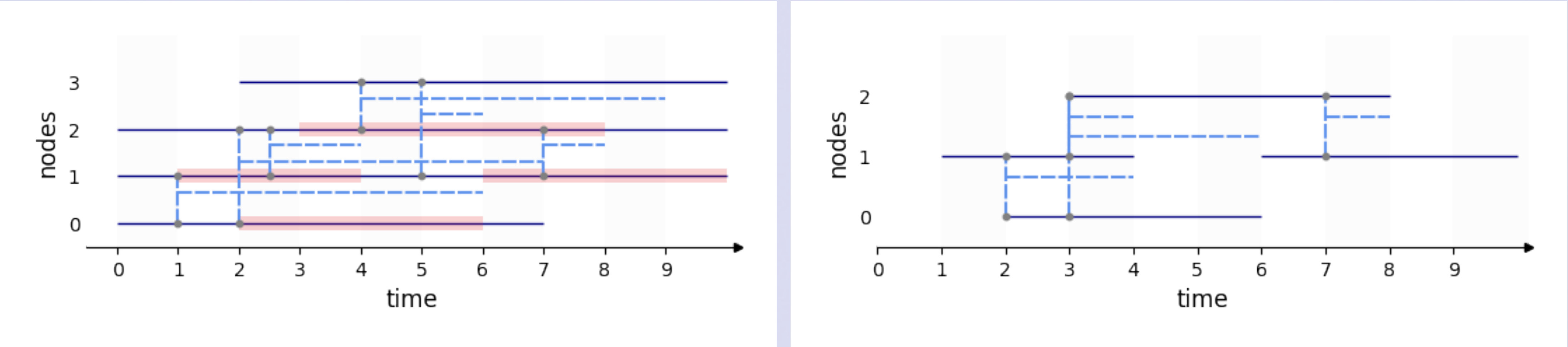


$$d = \frac{3}{6} = \frac{1}{2} \qquad d = \frac{3}{4} \qquad d = \frac{3}{3} = 1$$

STREAM GRAPHS

SG - Clusters & Substreams

In static networks, a cluster is a set of nodes, and we have defined an (induced) subgraph of this cluster as a graph composed of the nodes of the cluster and the edges existing between those nodes. In Stream Graphs, a clusters C is as subset of W, and the corresponding (induced) substream $S(C) = (T, V, C, E(C))$, with $E(C) = \{(t,(u,v)) \in E, (t,u),(t,v) \in C\}$.

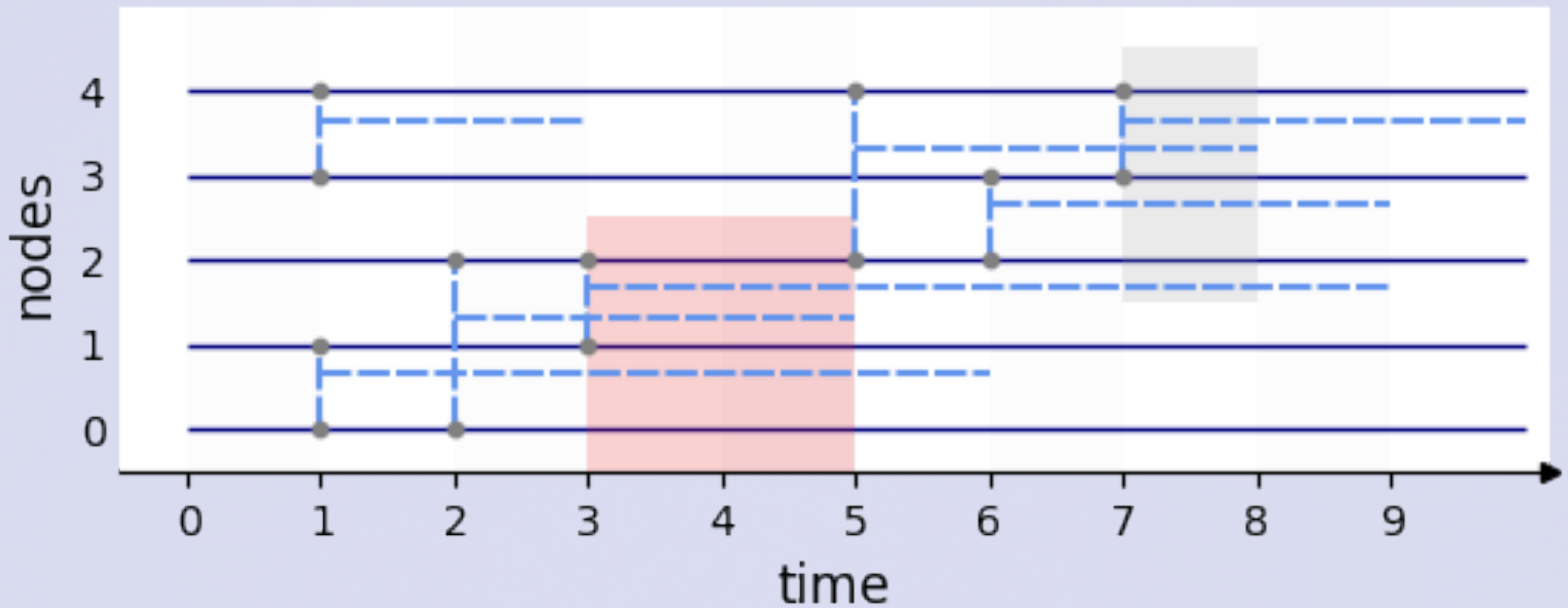


Example of subgraph (red,left) and induced substream (right).

STREAM GRAPHS

SG - Cliques

Having defined substreams and density, we can now naturally define a clique by analogy with static networks as a substream of density 1. A clique is said to be a **maximal clique** if it is not included in any other clique.



Red and Grey are the two maximal cliques of size three in this Stream Graph.

STREAM GRAPHS

SG - Neighborhood $N(u)$

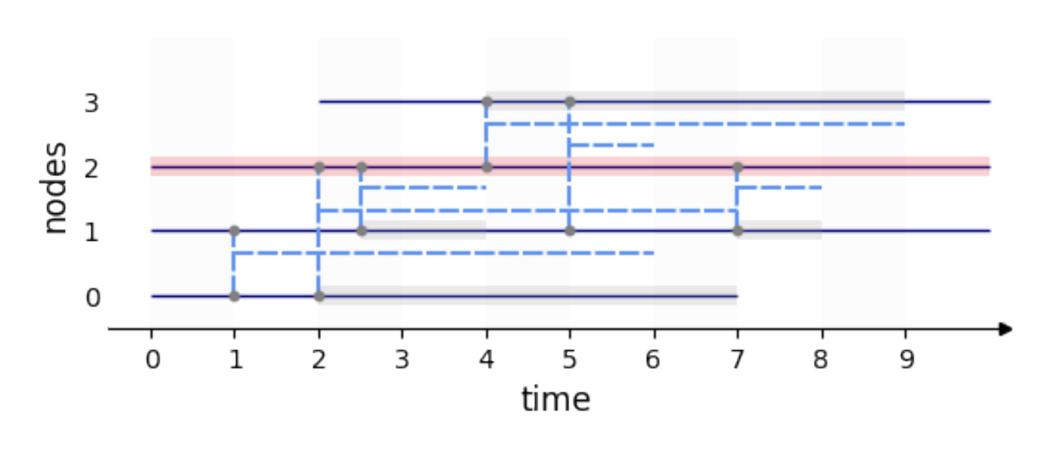
The neighborhood $N(u)$ of node u is defined as the cluster composed of node-times such as an edge-time exists between it and a node-time of u, i.e.,

$$N(u) = \{v_t, (u, v)_t \in E\}$$

SG - Degree $k(u)$

The degree $k(u)$ of node u is defined as the quantity of node in the Neighborhood of node u, i.e.

$$k(u) = |N(u)|$$



Example, the neighborhood of node 2 is highlighted in grey.
$k(c) = \frac{5+2.5+5}{10} = 1.25$.

STREAM GRAPHS

SG - Ego-network

The Ego network G_u of node u is defined as the substream induced by its neighborhood, i.e., $G_u = (T, V, N(u), E(N(u)))$.

SG - Clustering coefficient

The clustering coefficient $C(u)$ of node u is defined as the density of the ego-network of u, i.e.,

$$C(u) = d(N(u))$$

RANDOM MODELS FOR DYNAMIC NETWORKS

Laetitia Gauvin et al. “Randomized reference models for temporal networks”. In: *SIAM Review* 64.4 (Nov. 2022)

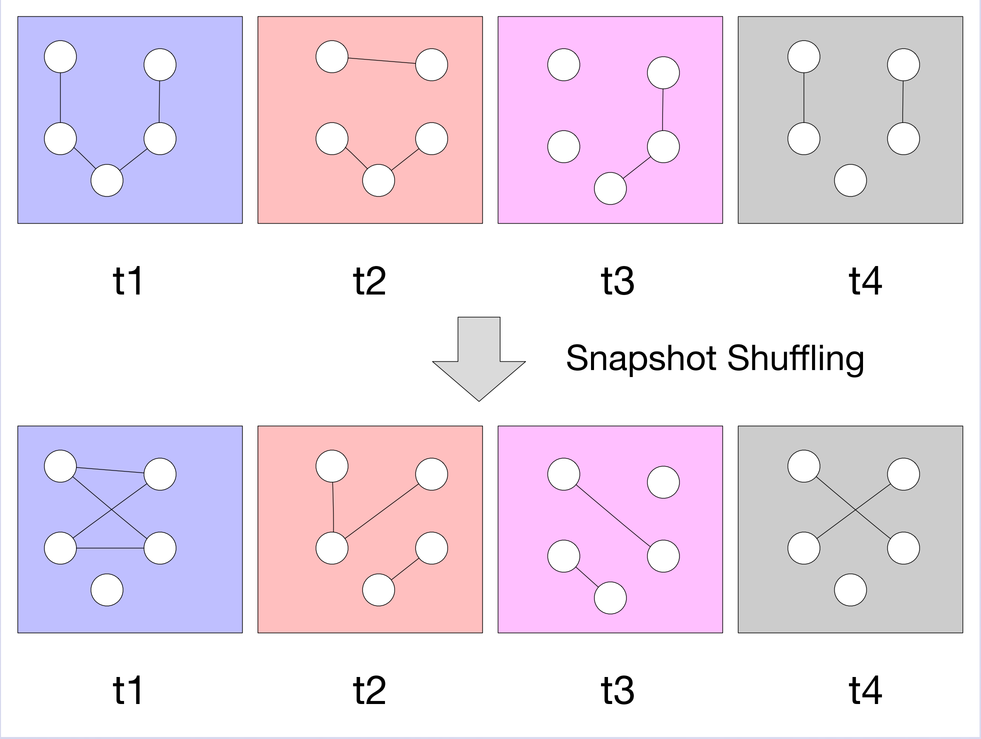
RANDOM MODELS

- 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

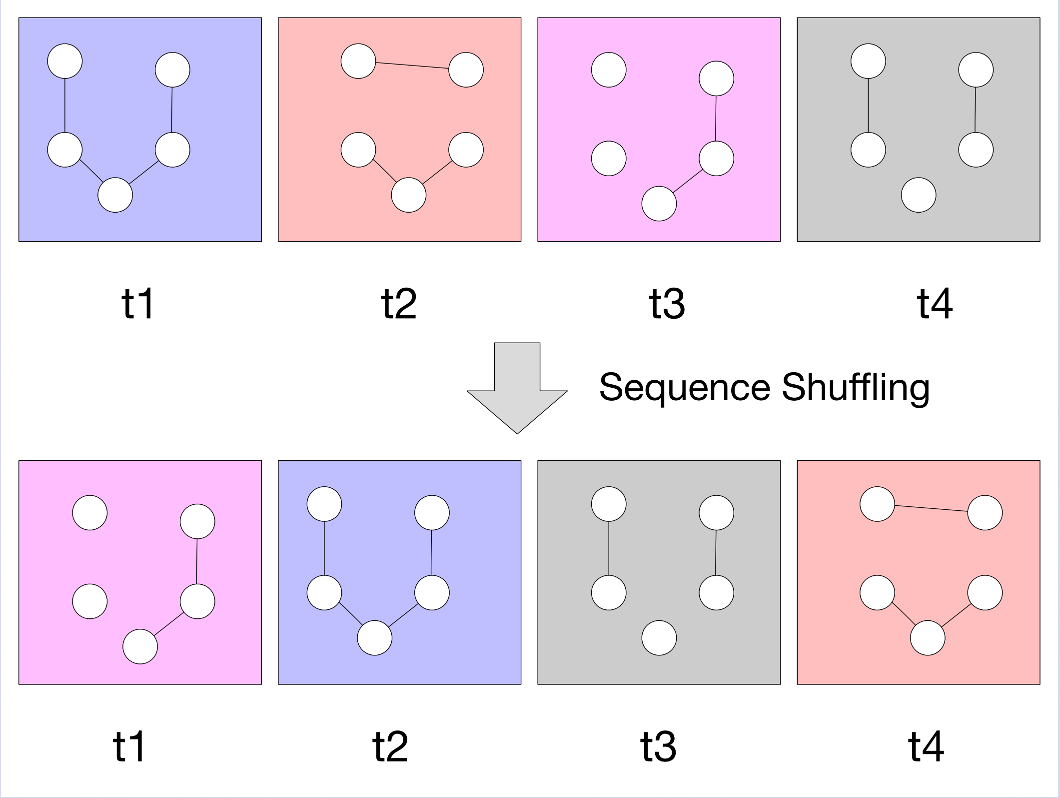
Snapshot Shuffling

Snapshot Shuffling keeps the order of snapshots, randomize edges inside snapshots. Any random model for static network can be used, such as ER random graphs or a degree preserving randomization.



Sequence Shuffling

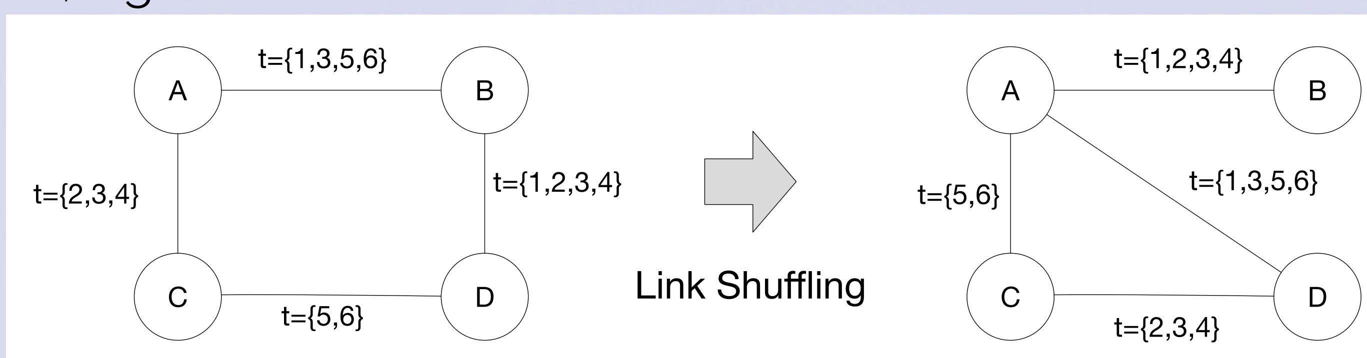
Sequence Shuffling keeps each snapshot identical, switch randomly their order.



RANDOM MODELS

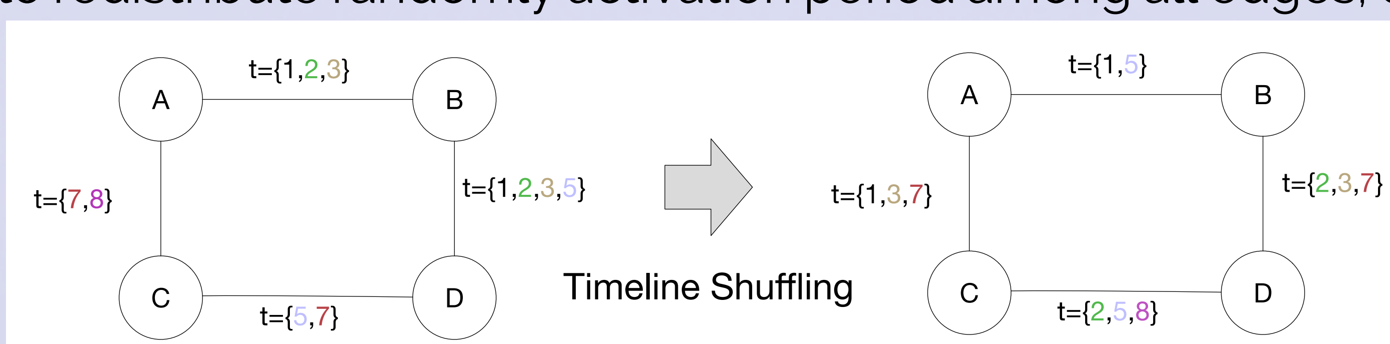
Link Shuffling

Link Shuffling keeps activation time per node pairs, randomize the aggregated graph. For instance, a simple way to achieve this is to pick two node pairs at random (connected or not) of the aggregated graph, and to exchange activation time of these node pairs, e.g.:



Timeline Shuffling

Timeline Shuffling keeps the aggregated graph, randomize edges activation time. For instance, a simple way to achieve this is to redistribute randomly activation period among all edges, e.g.:



DYNAMIC COMMUNITY DETECTION

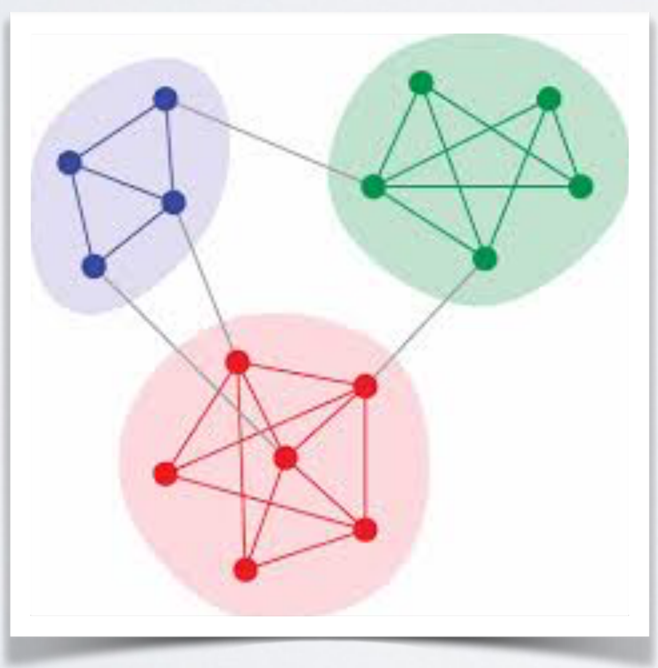
Rossetti, G., & Cazabet, R. (2018). Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, *51*(2), 1-37.

Cazabet, R., Boudebza, S., & Rossetti, G. (2020). Evaluating community detection algorithms for progressively evolving graphs. *Journal Of Complex Networks*

COMMUNITY DETECTION

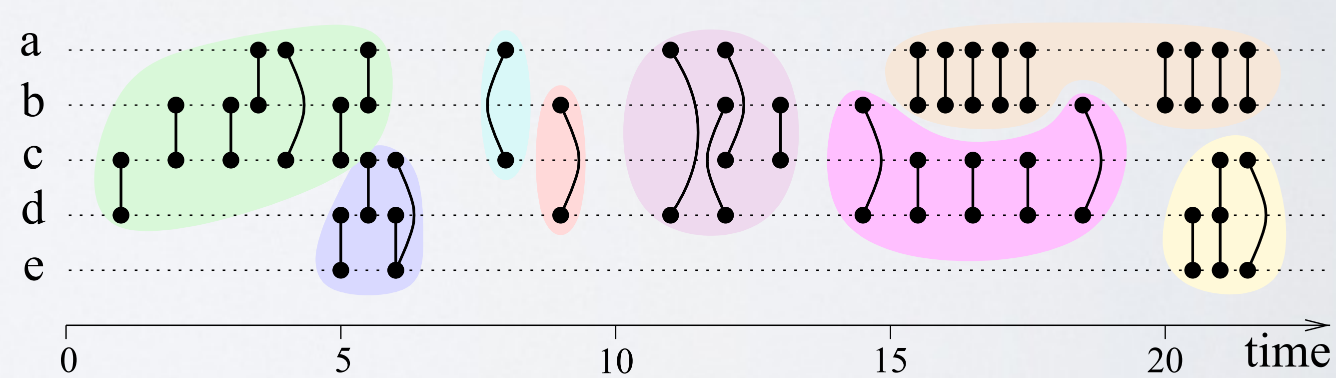
Static networks

Clusters: Sets of nodes



Dynamic Networks

Clusters: Sets of time-nodes, i.e., pairs (node,time)

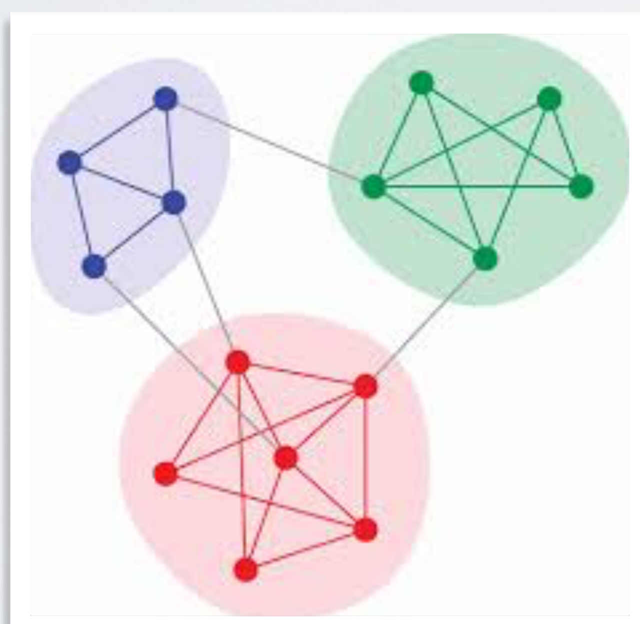


Gaumont, N., Viard, T., Fournier-S'Niehotta, R., Wang, Q., & Latapy, M. (2016). Analysis of the temporal and structural features of threads in a mailing-list. In *Complex Networks VII*

COMMUNITY DETECTION

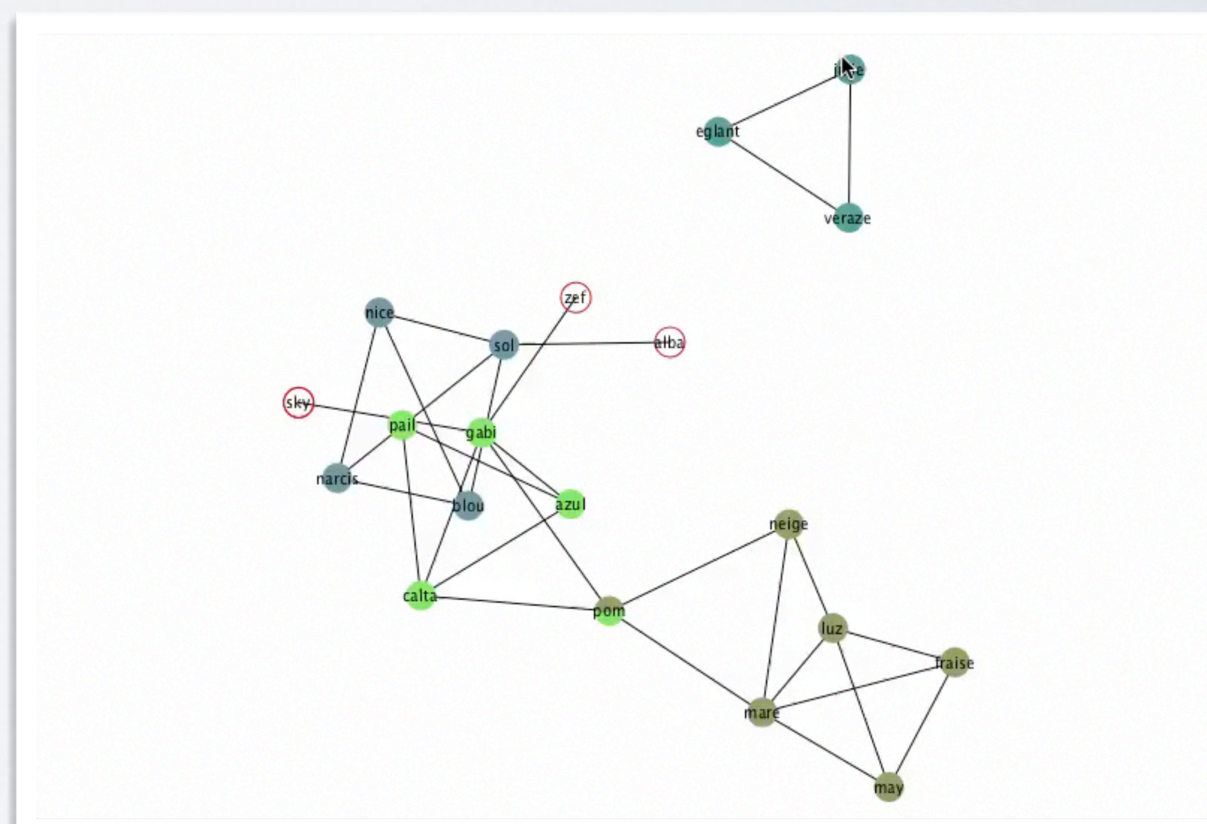
Static networks

Clusters: Sets of nodes



Dynamic Networks

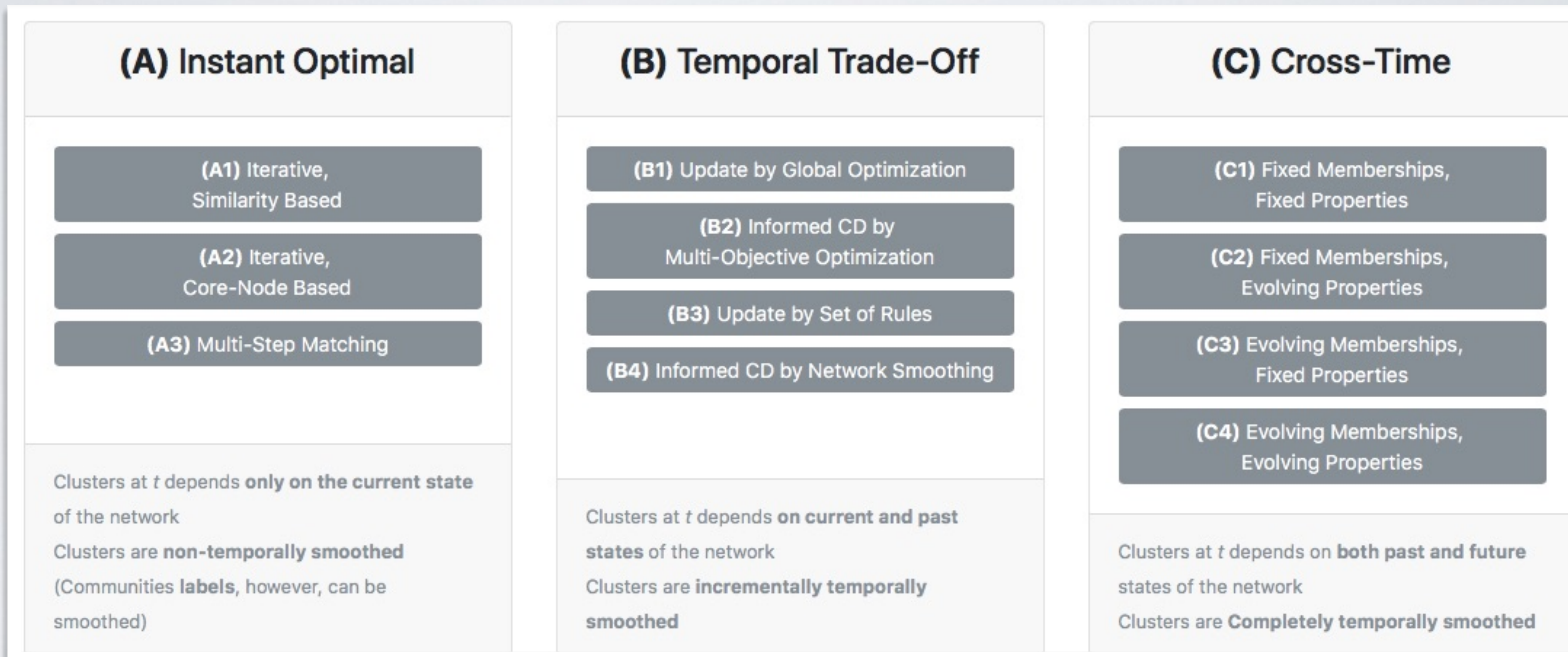
Clusters: Sets of time-nodes, i.e., pairs (node,time)



APPROACHES TO DCD

DYNAMIC COMMUNITIES ?

More than 50 methods published, broad categories



Rossetti, G., & Cazabet, R. (2018). Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, *51*(2), 1-37.

CATEGORIES

- Instant optimal:
  - Allows reusing static algorithms
  - No partition smoothing
  - Labels can be smoothed
  - Simple to parallelize

CATEGORIES

- Temporal trade-off
  - Cannot be parallelized (iterative)
  - => Best suited for real-time analysis / tasks
- Cross-Time
  - Requires to know the whole evolution in advance
  - => Not suited for real-time analysis, potentially the best smoothed (a posteriori interpretation)

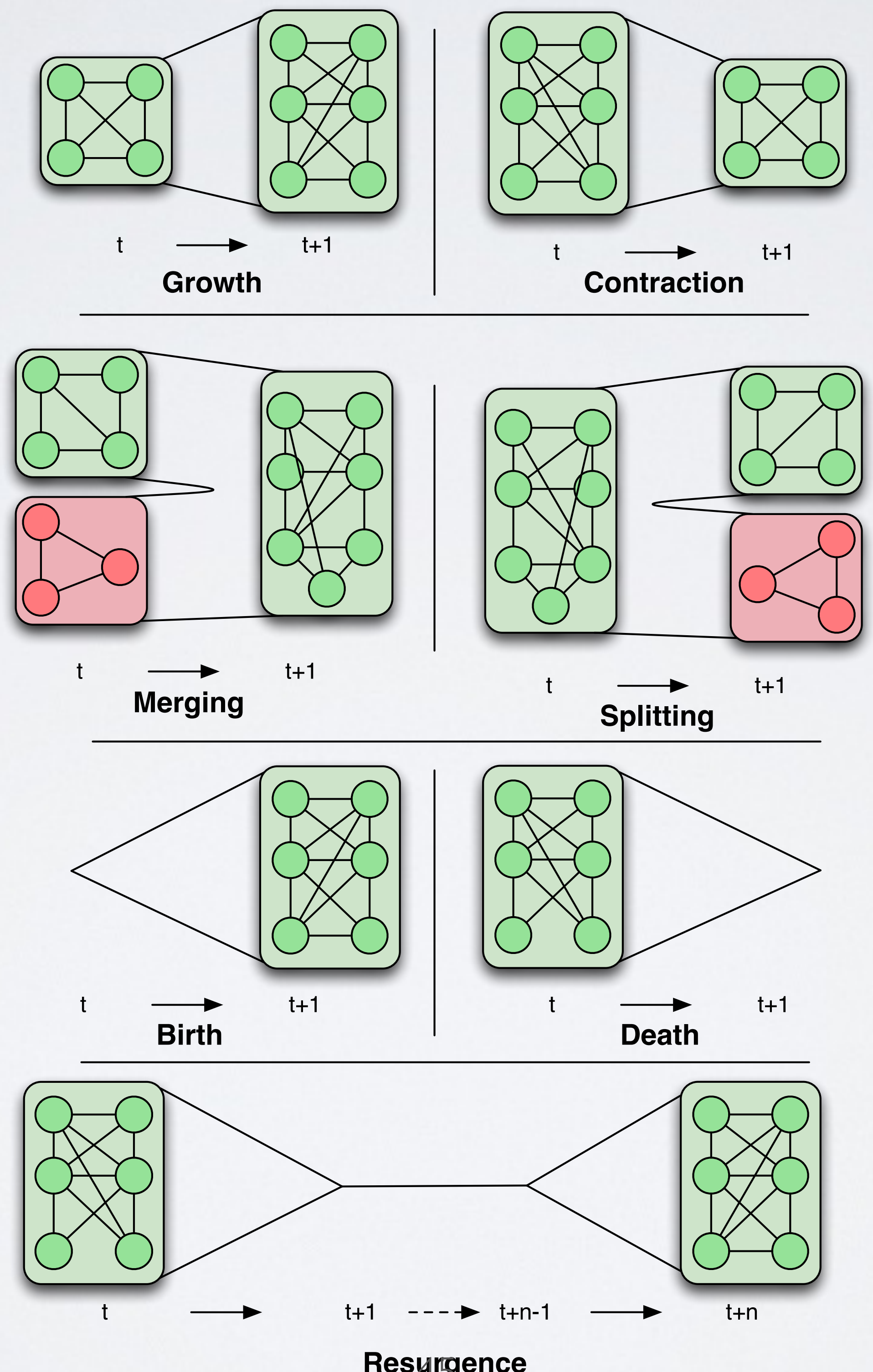
WHAT MAKES DCD INTERESTING

NARRATIVES ?

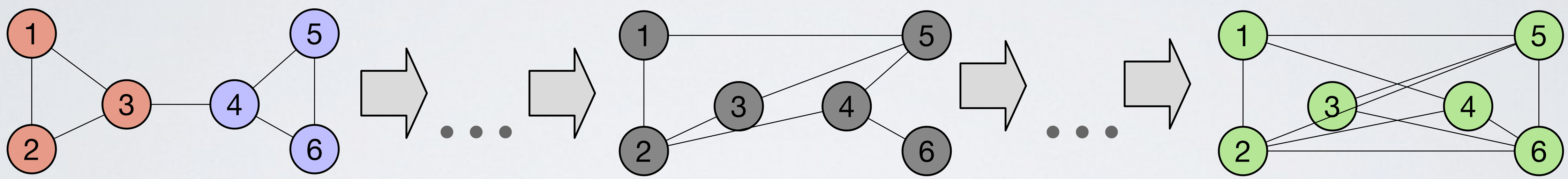
SMOOTHNESS / STABILITY

- No Smoothness: Partition at **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 EVENTS



PROGRESSIVE EVOLUTION



2 communities

??
Intermediate state

1 community

How to *track* communities, giving a *coherent* dynamic structure ?

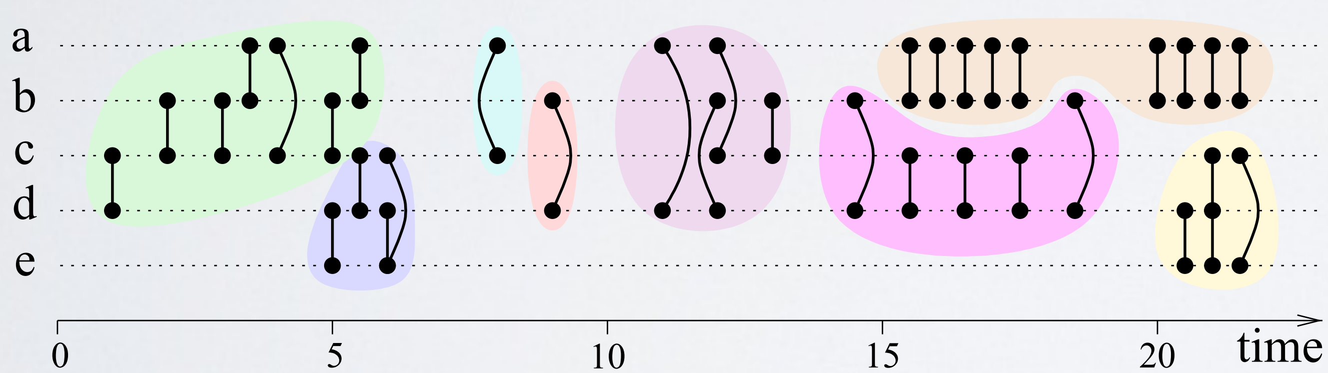
ONGOING WORK 1

How to adapt modularity for link streams?

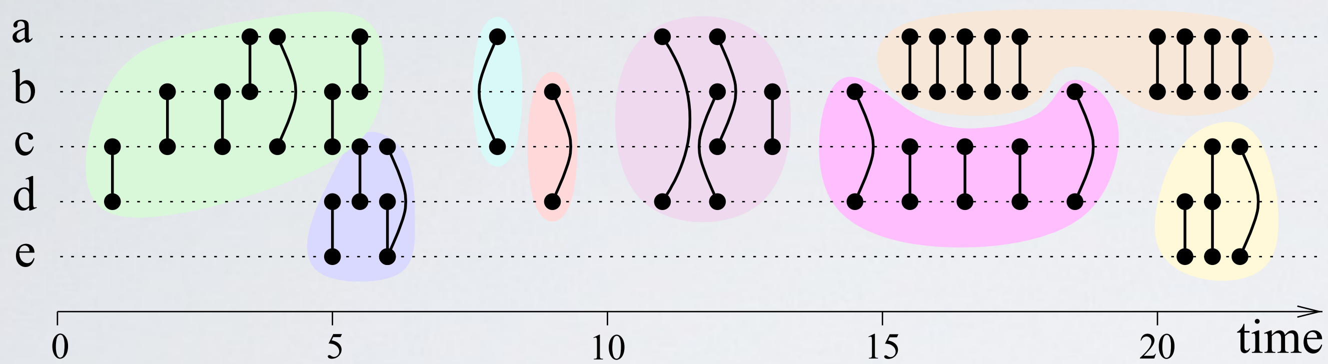
MODULARITY

- Most popular approach in static networks: modularity optimization
- Fraction of edges inside communities **-** Fraction of edges expected inside communities according to a **null model**

MODULARITY

- How to adapt for link streams ?
- 

LONGITUDINAL MODULARITY

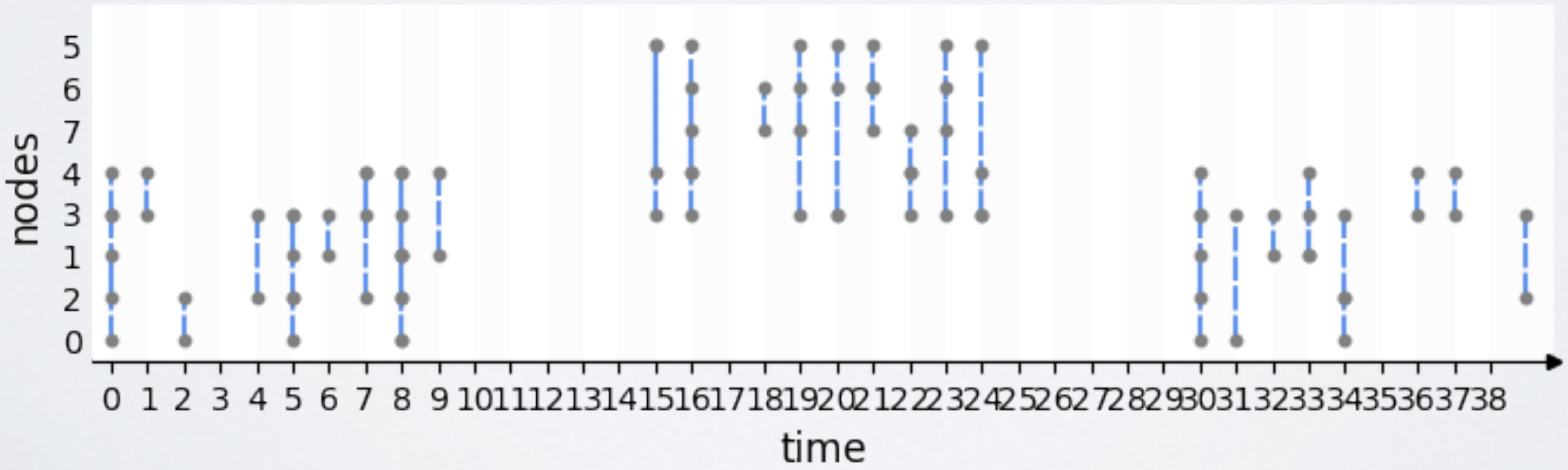
- 

- Fraction inside communities=> OK
- Fraction expected?
  - We need to choose a null model
    - Keep node degrees (whole period or locally ?) => Timeline Shuffling => Globally
    - How many links are expected between two nodes on an interval $[t_{start}, t_{end}]$?

$$\approx \frac{k_u k_v}{2m} \frac{[t_{start}, t_{end}]}{T}$$

LONGITUDINAL MODULARITY

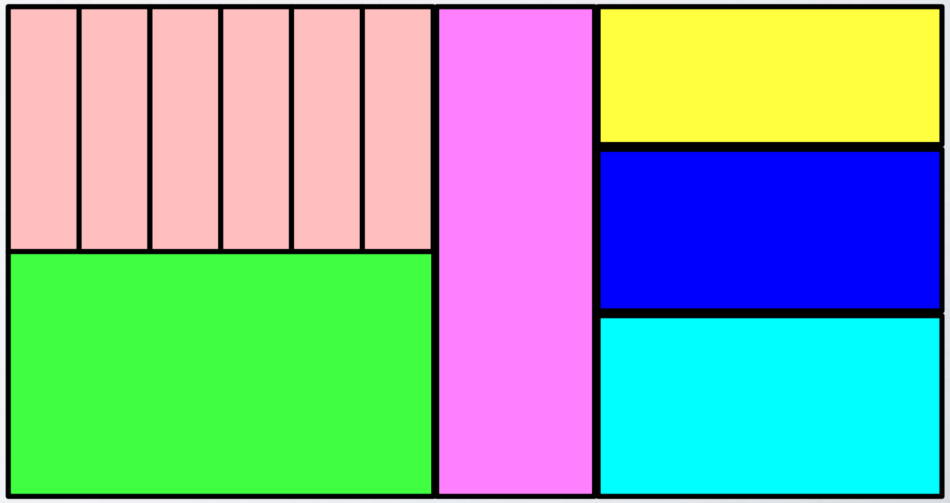
Intuitively correct: find communities where More links than expected by chance



LONGITUDINAL MODULARITY

- But… it does not work as expected

- Consider the red communities as a single community or separate ones
  - Number of edges inside does not change
  - Number of expected edges does not change
    - $$\approx \frac{k_u k_v}{2m} \frac{[t_{start}, t_{end}]}{T}$$



LONGITUDINAL MODULARITY

- The problem is a “smoothness” problem
  - No interest/gain to make a community last longer if it makes it less attractive at some point in time

- Solution Proposed: Work with edge repetitions
  - Modularity: Fraction of edges inside communities **-** Fraction of edges expected inside communities according to a null model
  - **Lmodularity:** Fraction of edges **repetitions** inside communities **-** Fraction of edges **repetitions** expected inside communities according to a null model

LONGITUDINAL MODULARITY

- Modularity works thanks to a trade-off between:
  - Each node added to a community allows to add some edges inside
    - Large gain for each edge (1)
    - Linear gain
  - Each node added to a community increases the potential number of edges
    - Small penalty for each edge (<<1)
    - **Quadratic** penalty (square of nodes inside the community)

- LM works in a similar way:
  - Making communities last longer allows to have more edge repetitions
    - Large gain for each edge
  - Making communities last longer increases **quadratically** the expectation of the number of repetitions

LONGITUDINAL MODULARITY

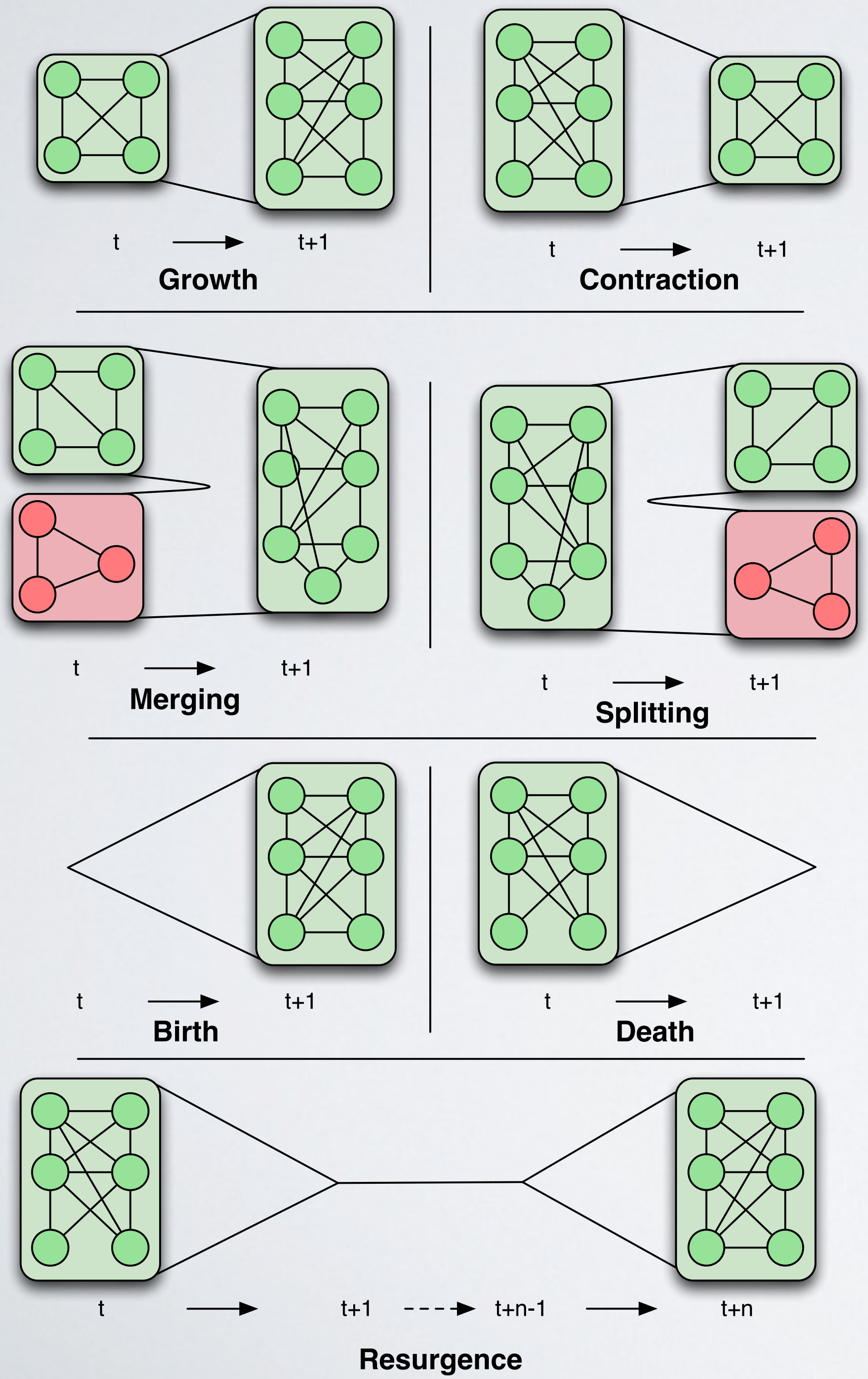
$$\textbf{Static:}\quad Q(A,\mathcal{C}) = \sum_{C\in\mathcal{C}} \sum_{i,j\in C^2} \left[\frac{A_{ij}}{2m} - \frac{k_i k_j}{4m^2}\right]$$

$$\textbf{Longitudinal:}\quad Q_{\mathcal{L}}(L,\mathcal{C}) = \sum_{C\in\mathcal{C}} \sum_{u,v\in V^2} \left[\frac{L^2_{uv\in C}}{2\mu} - \frac{\kappa_u \kappa_v}{4\mu^2} \frac{|T_{u\in C} \cap T_{v\in C}|^2}{|T|^2}\right]$$

ONGOING WORK 2

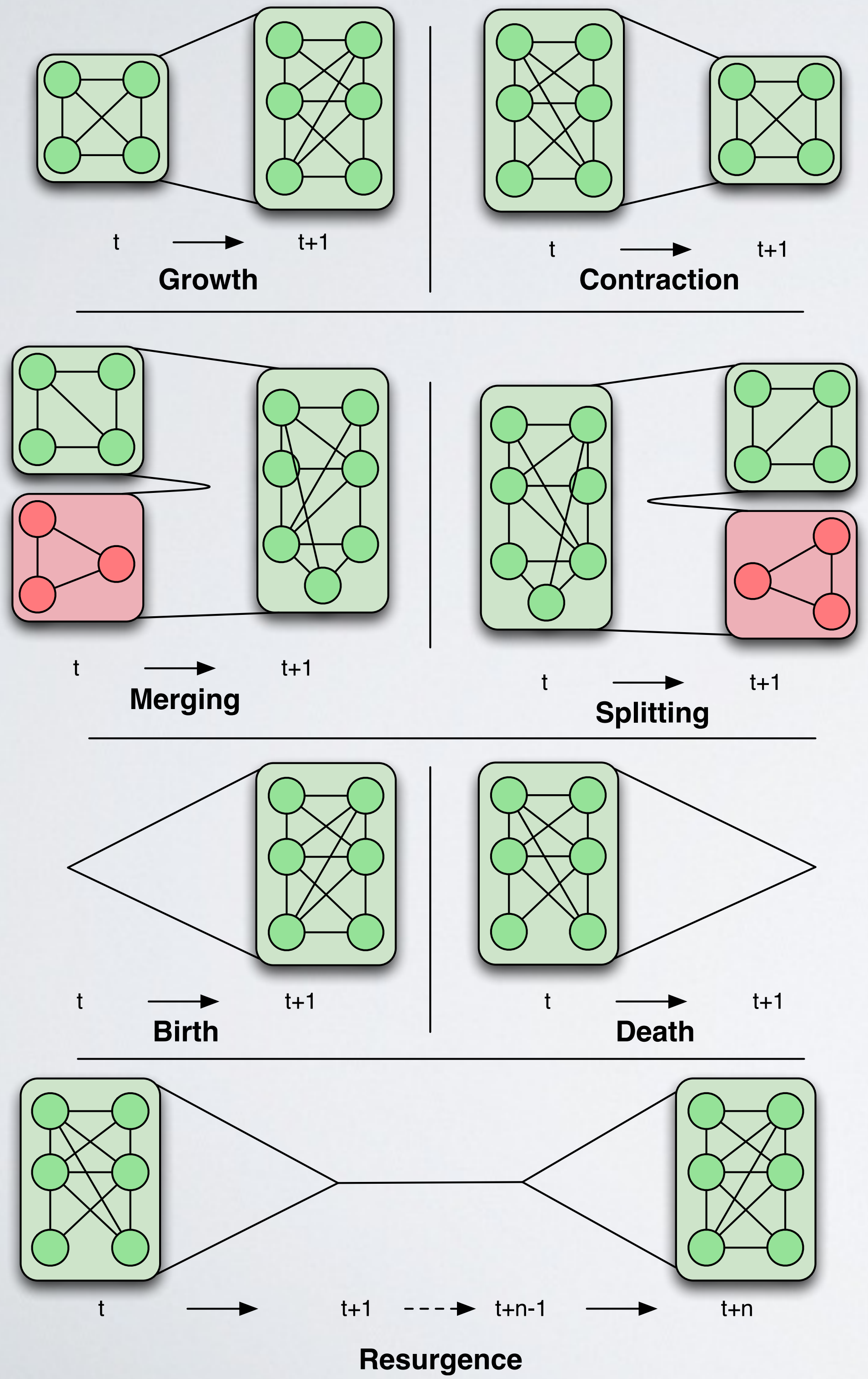
How to define community events quantitatively?

COMMUNITY EVENTS

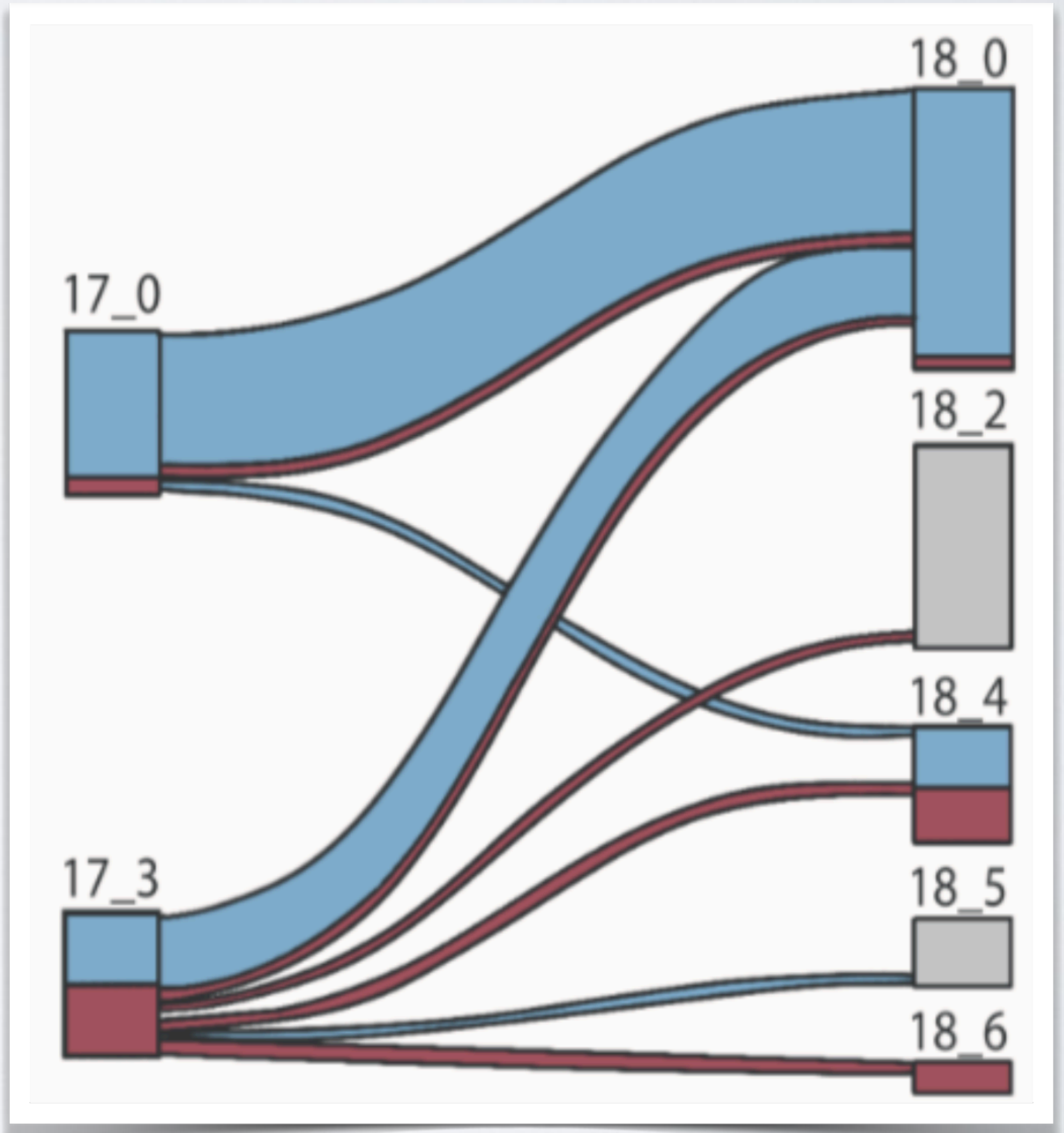


The beautiful theory

COMMUNITY EVENTS



The ugly truth



COMMUNITY EVENTS

- Given the successive partitions of a dynamic graph
  - How to decide what events take place
  - For instance to study quantitatively
    - Do I have many merges?
    - Many splits?
    - Are large communities splitting more than small ones?
    - etc.

COMMUNITY EVENTS

- Given the successive partitions of a dynamic graph
  - How to decide what events take place
  - For instance to study quantitatively
    - Do I have many merges?
    - Many splits?
    - Are large communities splitting more than small ones?
    - etc.
- Actually, the problem is not specific to communities in networks
  - Dynamic clustering

COMMUNITY EVENTS

Definition 1 (Flow Entropy)**.** *Let X be the target set, and $\mathcal{R} = \{R_1, \ldots, R_{|\mathcal{R}|}\}$ be the reference set. Let $B = \bigcup_{R \in \mathcal{R}} X \cap R$ identify the flow of X, namely the subset of X shared with any of the elements of $\mathcal{R}$. Let $\sigma(b)$ be a function that maps each $b \in B$ to a unique identifier indicating the set $R \in \mathcal{R}$ such that $b \in R$. The Flow Entropy is defined as:*

$$\mathcal{H} = \begin{cases} -\sum_{b \in B} \frac{p(\sigma(b)) \log_2 p(\sigma(b))}{\log_2 |\mathcal{R}|} & \textit{if } |\mathcal{R}| \geq 2 \\ 0 & o/w \end{cases} \tag{1}$$

The flow entropy quantifies the extent to which the nodes in X come from one or multiple sets at time $t-1$. The flow entropy is bounded in $[0, 1]$ due to the normalizing factor $\log_2 |\mathcal{R}|$. The more $\mathcal{H}$ approaches 0, the fewer sets contribute to X, and vice versa. However, two special cases need further discussion:

COMMUNITY EVENTS

Definition 2 (Contribution Factor)**.**

$$\mathcal{W} = \frac{1}{|X|} \sum_{R \in \mathcal{R}} |R \cap X| \frac{|R \cap X|}{|R|} \tag{2}$$

$\mathcal{W}$ measures the extent to which the target set is composed by the contributing sets from $t-1$ provide elements to X (respectively, the extent to which X provides elements

COMMUNITY EVENTS

Definition 3 (Difference Factor)**.**

$$\mathcal{D} = \frac{|X - \bigcup_{R \in \mathcal{R}} R|}{|X|} \tag{3}$$

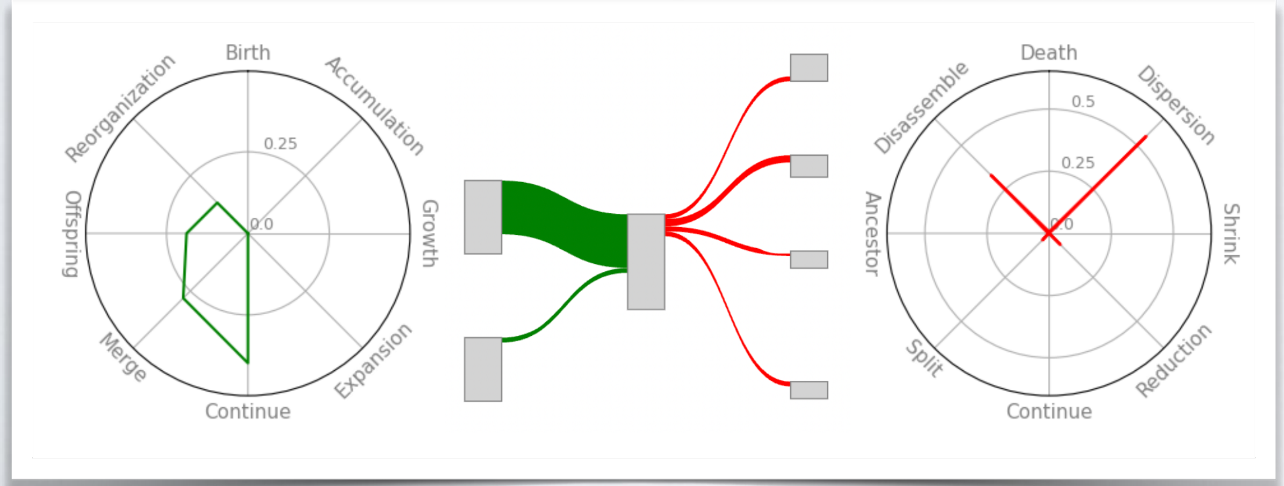
The difference factor quantifies the fraction of members in X that are not observed in the previous timestamp, i.e., the new, never-before-seen elements. In the remainder of this work, we will refer to these new elements as “joining” elements, as opposed to the elements belonging to the target set’s flow.

COMMUNITY EVENTS

Definition 5 (Backward Event Weights)**.** *Let X be the target set and $\mathcal{R}$ be the reference set such that X evolves from $\mathcal{R}$. Backward event weights quantify the extent to which X's evolution from $\mathcal{R}$ approximates one of the following transformations:*

$$\begin{aligned}
\textsc{Birth} &= (1-\mathcal{H})\cdot(1-\mathcal{W})\cdot\mathcal{D}\\
\textsc{Accumulation} &= \mathcal{H}\cdot(1-\mathcal{W})\cdot\mathcal{D}\\
\textsc{Continue} &= (1-\mathcal{H})\cdot\mathcal{W}\cdot(1-\mathcal{D})\\
\textsc{Merge} &= \mathcal{H}\cdot\mathcal{W}\cdot(1-\mathcal{D})\\
\textsc{Offspring} &= (1-\mathcal{H})\cdot(1-\mathcal{W})\cdot(1-\mathcal{D})\\
\textsc{Reorganization} &= \mathcal{H}\cdot(1-\mathcal{W})\cdot(1-\mathcal{D})\\
\textsc{Growth} &= (1-\mathcal{H})\cdot\mathcal{W}\cdot\mathcal{D}\\
\textsc{Expansion} &= \mathcal{H}\cdot\mathcal{W}\cdot\mathcal{D}
\end{aligned}$$

Definition 6 (Forward Event Weights)**.** *Let X be the target set and $\mathcal{R}$ be the reference set such that X evolves into $\mathcal{R}$. Forward event weights quantify the extent to which X's evolution into $\mathcal{R}$ approximates one of the following transformations:*



$$\begin{aligned}
\textsc{Death} &= (1-\mathcal{H})\cdot(1-\mathcal{W})\cdot\mathcal{D}\\
\textsc{Dispersion} &= \mathcal{H}\cdot(1-\mathcal{W})\cdot\mathcal{D}\\
\textsc{Continue} &= (1-\mathcal{H})\cdot\mathcal{W}\cdot(1-\mathcal{D})\\
\textsc{Split} &= \mathcal{H}\cdot\mathcal{W}\cdot(1-\mathcal{D})\\
\textsc{Ancestor} &= (1-\mathcal{H})\cdot(1-\mathcal{W})\cdot(1-\mathcal{D})\\
\textsc{Disassemble} &= \mathcal{H}\cdot(1-\mathcal{W})\cdot(1-\mathcal{D})\\
\textsc{Shrink} &= (1-\mathcal{H})\cdot\mathcal{W}\cdot\mathcal{D}\\
\textsc{Reduction} &= \mathcal{H}\cdot\mathcal{W}\cdot\mathcal{D}
\end{aligned}$$

COMMUNITY EVENTS