Network Science Cheatsheet

Made by
Remy Cazabet

Dynamic Networks

Disclaimer

Dynamic network analysis as introduced here is a recent and still not fully mature field, with a large number of contributions, for which we cannot know yet which one will stand the test of time. This is therefore *my* vision of the dynamic network field *as of today*.

Ubiquity of Dynamic Networks

Most real networks are in fact dynamic: nodes and edges appear and disappear with time. Think of friendship in social networks, flight routes or any human interactions. Networks are often analyzed as static objects because 1)it's harder to obtain dynamic information, 2)Taking dynamic into account makes some analysis more difficult.
While more and more aspects of our life become linked to digital technology, datasets with fine temporal information also become more and more common.

Snapshots & Aggregated Networks

Static networks representing dynamic information can be obtained by two processes:

- **Snapshots** correspond to the state of a network at a particular point in time, e.g., all follower/followees relationship at a particular second
- **Aggregated Networks** are obtained by cumulating information over a period of time, e.g., in a phone call network, in the snapshot representing year 2020, an edge exists between two individuals if they called each other at least once over the year 2020.

Interactions or Relation?

Dynamic networks can be used to represent different types of real data. In particular, they can be used to represent networks of **relations** and networks of *interactions*. For instance, friendships, acquaintances, physical wires, roads, etc. can be thought as *relations*, while e-mails, phone calls, instant messages, physical contacts, etc. are *interactions*.
There is often a relation between these two notions: interactions tend to occur between entities having a relation, and/or relations tend to form between entities having interactions.

Dynamic Network Properties

Independently of the studied data, dynamic networks can have various properties:

- **Edge** presence can be **punctual** or **with duration**
- **Node** presence can be **unspecified**, **punctual** or **continuous**
- If **time is continuous**, it can be **bounded** on a period of analysis or **ubounded**
- If **nodes** have attributes, they can be **constant** or **time-dependent**
- If **edges** have weights, they can be **constant** or **time-dependent**

Vocabulary

Many different names have been used to for networks changing with time, but there is no broad consensus in the literature on the meaning of those terms, unless they are used with an explicit reference to a paper defining those terms. Here is a list of the most popular:

- **Dynamic Networks** and **Dynamic Graphs**
- **Longitudinal Networks**
- **Evolving Graphs**
- **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)

Slowly Evolving/Degenerate

Beyond the nature of the data and the chosen representation, a critical difference defining how a dynamic network can be analyzed is whether it is a **Slowly Evolving Network (SEN)** or **Degenerate**. In a SEN network, to each instant corresponds a well defined graph, that can be studied with usual tools of network science. In a degenerate temporal network, a meaningful graph can be obtained only when aggregating it over a period Δ.

Analyzing SEN

A slowly evolving network can easily be studied by the tools already defined on static graphs. For any instant (discrete or continuous), one can compute network descriptors (density, clustering coefficient, etc.), node descriptors (centralities), reachability, etc.

Analyzing degenerate networks

A degenerate network can always be transformed into a SEN by aggregating it using time windows, fixed (yielding snapshots, i.e., discrete SEN) or sliding (yielding continuous SEN). But a more powerful solution is to study them in their original form, without loosing any temporal information through aggregation. This however requires new definitions.

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.

SG - Time-Entity designation

Stream Graphs 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 exist if node v is present at time t
$(u,v)_t$	**Edge-time**: $(u,v)_t$ exist if edge (u,v) is present at time t, 0 otherwise
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

SG - Node/Edge presence

Nodes and Edges are typically present in the graph only for a fraction of its total duration, Node/Edge presence is computed as the fraction of the total times during which it is present. Note that if time is continuous and edges are discrete, we take by convention $|T| = 1$, i.e., we simply count nodes/edges presence time.

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 $\frac{\lvert T_{uv} \rvert}{\lvert T \rvert}$

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.

SG - N

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.

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.

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

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

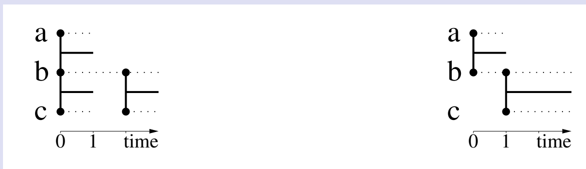
- Ignoring nodes duration: $L^1_{\max} = |V|^2$
- Ignoring nodes co-presence $L^2_{\max} = N^2$
- Taking nodes co-presence into account: ${L_{\max}}^3 = \sum_{(u,v),u,v \in V} |T_u \bigcap T_v|$

SG - Density - d

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

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

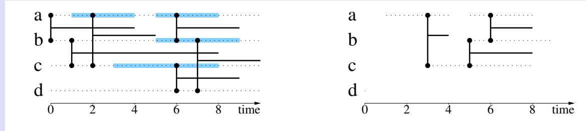
The definition can naturally be extended by using the definitions of L and $L_{\max}$ introduced on Stream Graph. In (Latapy, Viard, and Magnien 2018), the authors use ${L_{\max}}^3$. This definition can also be understood as the probability, if we take a time at random, and two nodes alive a that time at random, for them to be connected. Note that a common way to define the density in static networks is $d = N^2$, because N^2 is the only way to define $L_{\max}$ in static networks, unlike in Stream Graphs.



Examples of graphs with $N = 2$ nodes, $L = 1$ link, but with different densities, respectively 0.75 (left) and 1 (right).

SG - Clusters & Substreams

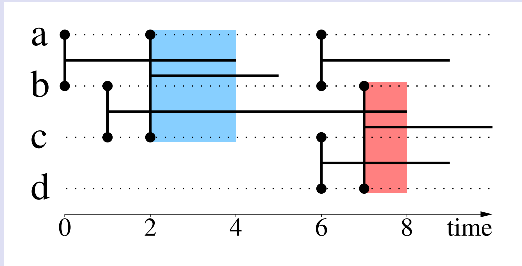
In static networks, a cluster is a set of nodes, and we have defined an (induced) subgraph of this cluster as a graph composed of the nodes of the cluster and the edges existing between those nodes. In Stream Graphs, a clusters C is as subset of W, and the corresponding (induced) substream $S(C) = (T,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



Example of subgraph and induced substream.

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 Blue are the two maximal cliques of size three in this Stream Graph.

SG - Neighborhood $N(u)$

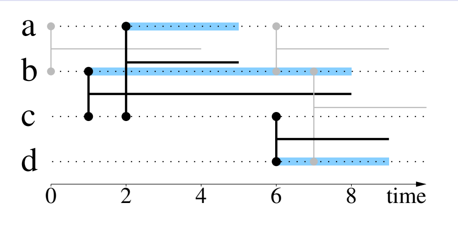
The neighborhood $N(u)$ of node u is defined as the cluster composed of node-times such as an edge-time exists between it and a node-time of u, i.e.,

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

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 c is highlighted in blue.
$k(c) = 1.3$
(|[1,8]|+|[2,5]|+|[6,9]|).

SG - Ego-network

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

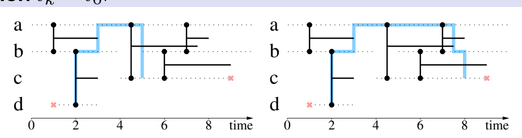
SG - Clustering coefficient

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

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

SG - Paths

In a Stream Graph S=(T,V,W,E), a **path** P from node-time x_α to node-time y_ω is a sequence $(t_0, x, v_0), (t_1, v_0, v_1), ..., (t_k, v_k, y)$ of elements of $T \times V \times V$ such that $t_0 \geq \alpha$,$t_k \leq \omega$, $((t_i, u_i, v_i)) \in E$. We say that P **starts at** t_0, **arrives at** t_k, has **length** $k+1$ and **duration** $t_k - t_0$.



Examples of two paths. The left one starts at 2, arrives at 5, has length 3 and duration 3. The right one starts at 2, arrives at 7.5, has length 4 and duration 5.5.

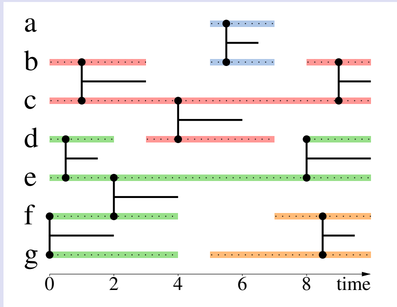
SG - Shortest - Fastest - Foremost

- **Shortest Paths**, as in static networks, are paths of **minimal length**.
- **Fastest Paths** are paths of **minimal duration**.
- **Foremost Paths** are paths **arriving first**.

Furthermore, one can combine those properties, defining for instance:
Fastest shortest paths (paths of minimum duration among those of minimal length)
Shortest fastest paths (paths of minimal length among those of minimal duration)

SG - Connected Components

Various definitions for connected components have been proposed for temporal networks, see (Latapy, Viard, and Magnien 2018) for details. One of the simplest one is the **weakly connected component**, defined such as two node-times belong to the same connected component if and only if there is a path from one to the other, *ignoring time*.



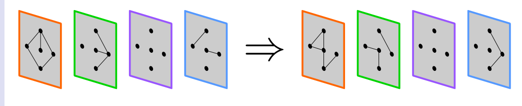
Example of a Stream Graph decomposed in 4 weakly connected components.

Random Models

We have seen that comparing an observed network with a randomized version of it has many applications. In dynamic networks, many variants have been proposed. In (Gauvin et al. 2018), the authors consider methods defined on sequences of snapshots that conserve nodes and number of events, and grouped them in 4 main families, **Snapshot Shuffling, Sequence Shuffling, Link Shuffling** and **Timeline Shuffling**.

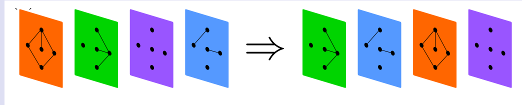
Snapshot Shuffling

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 the Configuration Model.



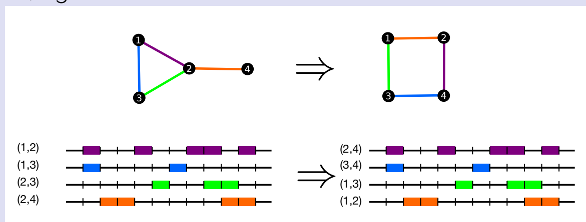
Sequence Shuffling

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



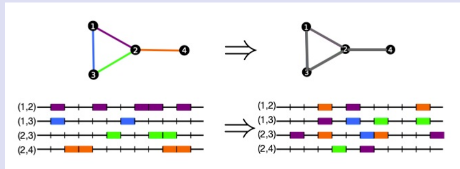
Link Shuffling

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.:



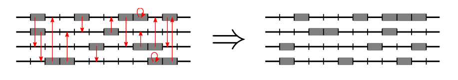
Timeline Shuffling

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.:



More constrained Shuffling

Variants of these shufflings with additional constraints have been proposed, for instance the **Local timeline shuffling**, randomizing events time edge by edge, or the **Weight constrained timeline shuffling**, randomizing events while conserving the number of observations for each edge. See (Gauvin et al. 2018) for more.



References

[1] Arnaud Casteigts et al. "Time-varying graphs and dynamic networks". In: *International Journal of Parallel, Emergent and Distributed Systems* 27.5 (2012), pp. 387–408.

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

[3] Petter Holme and Jari Saramäki. "Temporal networks". In: *Physics reports* 519.3 (2012), pp. 97–125.

[4] 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.