Dynamics and Multiscale Modular Structure in Networks

R. Lambiotte
Institute for Mathematical Sciences
Imperial College London

with M. Barahona and J.-C. Delvenne

R. Lambiotte, J.-C. Delvenne and M. Barahona, arXiv:0812:1770
1. Modules and Hierarchies
2. Quality of a partition: modularity and stability
3. Modularity optimisation
4. Selection of the most relevant scales
Modular Networks

Most networks are very inhomogeneous and are made of modules: many links within modules and a few links between different modules.

- Internet
- Power grids
- Food webs
- Metabolic networks
- Social networks
- The brain
- Etc.
Modular Networks

Networks have a hierarchical structure: modules within modules

Uncovering communities/modules helps to understand the structure of the network, to uncover similar nodes and to draw a readable map of the network (when N is large).
Modular Networks and dynamics

Many networks are “modular” and have a hierarchical structure: modules within modules

How does such modularity affect dynamics?


Is it possible to uncover those modules in large networks?

*NG, GN, Walktrap, clique-percolation, Simulated Annealing, etc.*
Modular Networks and dynamics

Many networks are “modular” and have a hierarchical structure: modules within modules

How does such modularity affect dynamics?


Is it possible to use dynamics to characterize (and uncover?) the modular structure of a network?
*e.g. Walktrap (RW exploration), Rosvall and Bergstrom, etc.*

Is it possible to uncover those modules in large networks?
*NG, GN, Walktrap, clique-percolation, Simulated Annealing, etc.*
1. Modules and Hierarchies

2. Quality of a partition: modularity and stability

3. Modularity optimisation

4. Selection of the most relevant scales
Notations

Let us focus on an unweighted, undirected network

\[ A_{ij} \quad \text{adjacency matrix} \]

\[ k_i = \sum_j A_{ij} \quad \text{degree of } i \]

\[ m = \frac{1}{2} \sum_i k_i \quad \text{total number of links} \]
Quality of a partition

What is the best partition of a network into modules?
Modularity

Q = fraction of edges within communities - expected fraction of such edges

Let us attribute each node \( i \) to a community \( c_i \)

\[
Q = \frac{1}{2m} \sum_{i,j} \left[ A_{ij} - P_{ij} \right] \delta(c_i, c_j) \quad Q \in [-1, 1]
\]

\( P_{ij} = \frac{k_i k_j}{2m} \) expected number of links between \( i \) and \( j \)

\[
Q_C = \frac{1}{2m} \sum_{i,j} \left[ A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)
\]
Modularity

Optimising modularity uncovers one partition

What about sub (or hyper)-communities in a hierarchical network?

Reichardt & Bornholdt

$$Q_{\gamma} = \frac{1}{2m} \sum_{i,j} \left[ A_{ij} - \gamma P_{ij} \right] \delta(c_i, c_j)$$

Tuning parameters allow to uncover communities of different sizes

Reichardt & Bornholdt different of Arenas, except in the case of a regular graph where

$$\gamma = 1 + \frac{r}{\langle k \rangle}$$


Stability

The quality of a partition is determined by the patterns of a flow within the network: a flow should be trapped for long time periods within a community before escaping it.

The stability of a partition is defined by the statistical properties of a random walker moving on the graph:

Stability

The quality of a partition is determined by the patterns of a flow within the network: a flow should be trapped for long time periods within a community before escaping it.

The stability of a partition is defined by the statistical properties of a random walker moving on the graph:

\[
R(t) = \sum_{C \in \mathcal{P}} P(C, t_0, t_0 + t) - P(C, t_0, \infty)
\]

\[P(C, t_0, t_0 + t)\] probability for a walker to be in the same community at times \(t_0\) and \(t_0 + t\) when the system is at equilibrium

\[P(C, t_0, \infty)\] probability for two independent walkers to be in \(C\) (ergodicity)

Stability

Let us consider a continuous-time random walk with Poisson waiting times

\[ \dot{p}_i = \sum_j \frac{A_{ij}}{k_j} p_j - p_i \]

\[ p^*_i = \frac{k_i}{2m} \]

\[ R(t) = \sum_{i,j} \left[ \left( e^{t(B-I)} \right)_{ij} \frac{k_j}{2m} - \frac{k_i k_j}{(2m)^2} \right] \delta(c_i, c_j) \]

\[ B_{ij} = \frac{A_{ij}}{k_j} \]

Probability that a walker is in the same community initially and at time t

Same probability for independent walkers

R. Lambiotte, J.-C. Delvenne and M. Barahona, arXiv:0812:1770
Stability: time as a resolution parameter

What are the optimal partitions of \( R_t \)?

\[
R(0) = 1 - \sum_{i,j} \frac{k_i k_j}{(2m)^2} \delta(c_i, c_j)
\]

Communities=single nodes

\[
R(t) \approx (1 - t)R(0) + tQ_C \equiv Q(t)
\]

favours single nodes  modularity

!! \( Q_t \) equivalent to the Hamiltonian formulation of Reichardt and Bornholdt (t=1/\gamma)

When \( t \) goes to infinity, the optimal partition is made of 2 communities (by spectral decomposition)
Stability: time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased

![Diagram showing stability over time with different resolutions](image)
Stability: time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased
Time is a “resolution parameter”: larger and larger communities when time is increased.
Time is a “resolution parameter”: larger and larger communities when time is increased.
Time is a “resolution parameter”: larger and larger communities when time is increased
Important observation:

The stability $R(t)$ of the partition of a graph with adjacency matrix $A$ is equivalent to the modularity $Q$ of a time-dependent graph with adjacency matrix $X(t)$

$$X_{ij}(t) = (e^{t(B-I)})_{ij} k_j$$

which is the flux of probability between 2 nodes at equilibrium and whose generalised degree is

$$\sum_j X_{ij}(t) = k_i$$

$$R(t) = \sum_{i,j} X_{ij}(t) / 2m - k_i k_j / (2m)^2 \delta(c_i, c_j) = Q(X(t))$$

For very large networks:  

$$R(t) \approx (1 - t)R(0) + tQ_C \equiv Q(t)$$
1. Modules and Hierarchies
2. Quality of a partition: modularity and stability
3. Modularity optimisation
4. Selection of the most relevant scales
Modularity optimisation

Different types of algorithm for different applications:

Small networks (<$10^3$): Simulated Annealing
Intermediate size ($10^3$ - $10^4$): Spectral methods, PL, etc.
Large size (> $10^5$): greedy algorithms
Our algorithm

The algorithm is based on two steps that are repeated iteratively.

First phase: Find a local maximum

1) Give an order to the nodes (0,1,2,3,...., N-1)

2) Initially, each node belongs to its own community (N nodes and N communities)

3) One looks through all the nodes (from 0 to N-1) in an ordered way. The selected node looks among its neighbours and adopt the community of the neighbour for which the increase of the quality function is maximum (and positive).

4) This step is performed iteratively until a local maximum of modularity is reached (each node may be considered several times).

Our algorithm

Once a local maximum has been attained, we build a new network whose nodes are the communities. The weight of the links between communities is the total weight of the links between the nodes of these communities.

New network of 4 nodes!

Note the self-loops

In typical realizations, the number of nodes diminishes drastically at this step.
Our algorithm

The two steps are repeated iteratively, thereby leading to a hierarchical decomposition of the network.
Advantages of the Louvain method

Known to perform very well for optimising modularity:

<table>
<thead>
<tr>
<th></th>
<th>Karate</th>
<th>Arxiv</th>
<th>Internet</th>
<th>Web nd.edu</th>
<th>Phone</th>
<th>Web uk-2005</th>
<th>Web</th>
<th>WebBase 2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes/links</td>
<td>34/77</td>
<td>9k/24k</td>
<td>70k/351k</td>
<td>325k/1M</td>
<td>2.04M/5.4M</td>
<td>39M/783M</td>
<td>118M/1B</td>
<td></td>
</tr>
<tr>
<td>CNM</td>
<td>.38/0s</td>
<td>.772/3.6s</td>
<td>.692/799s</td>
<td>.927/5034s</td>
<td>-/-</td>
<td>-/-</td>
<td>-/-</td>
<td></td>
</tr>
<tr>
<td>PL</td>
<td>.42/0s</td>
<td>.757/3.3s</td>
<td>.729/575s</td>
<td>.895/6666s</td>
<td>-/-</td>
<td>-/-</td>
<td>-/-</td>
<td></td>
</tr>
<tr>
<td>WT</td>
<td>.42/0s</td>
<td>.761/0.7s</td>
<td>.667/62s</td>
<td>.898/248s</td>
<td>.553/367s</td>
<td>-/-</td>
<td>-/-</td>
<td></td>
</tr>
<tr>
<td>Our algorithm</td>
<td>.42/0s</td>
<td>.813/0s</td>
<td>.781/1s</td>
<td>.935/3s</td>
<td>.76/44s</td>
<td>.979/738s</td>
<td>.984/152mn</td>
<td></td>
</tr>
</tbody>
</table>

Also works quite well for $Q(t)$. $R(t) \approx (1 - t)R(0) + tQ_C \equiv Q(t)$

E.g., football network:

```
SA, iteration factor=1.0; cooling factor=0.995
0.0 115 0.991245 115 23
0.1 115 0.894656 33 63
0.2 115 0.850513 14 58
0.3 115 0.817482 12 40
0.4 115 0.786487 12 36
0.5 115 0.755492 12 36
0.6 115 0.724497 12 36
0.7 115 0.693602 11 36
0.8 115 0.663422 11 33
0.9 115 0.633770 10 58
1.0 115 0.601357 9 79
```

Louvain method:

```
0.0 115 0.991245 115 0
0.1 115 0.894616 33 0
0.2 115 0.850513 14 0
0.3 115 0.817482 12 0
0.4 115 0.786487 12 0
0.5 115 0.755492 12 0
0.6 115 0.724497 12 0
0.7 115 0.693602 11 0
0.8 115 0.663422 11 0
0.9 115 0.633792 10 0
1.0 115 0.604570 10 0
```
1. Modules and Hierarchies
2. Quality of a partition: modularity and stability
3. Modularity optimisation
4. Selection of the most relevant scales
The optimization of $R(t)$ over a period of time leads to a sequence of partitions that are optimal at different time scales. How to select the most relevant scales of description?

The significance of a particular scale is usually associated to a certain notion of the robustness of the optimal partition. Here, robustness indicates that a small modification of the optimization algorithm, of the network, or of the quality function does not alter this partition.

We look for regions of time where the optimal partitions are very similar. The similarity between two partition is measured by the normalised variation of information.
**Conclusion**

- Relation between dynamics and the hierarchical structure of networks
- Dynamical formulation for the quality of a partition
- Changing time allows to zoom in and out
- Algorithms developed in order to optimise stability/modularity very large networks

- Different dynamics lead to different quality functions for the partition of a graph
- Modularity and Stability are radically different in the case of directed networks

Original Louvain method to optimise modularity available on [http://findcommunities.googlepages.com](http://findcommunities.googlepages.com)

Generalized codes to optimise $Q_t$ available on [http://www.lambiotte.be](http://www.lambiotte.be)

Thanks to J.-L. Guillaume (for providing his c++ code)

R. Lambiotte, J.-C. Delvenne and M. Barahona, *arXiv:0812:1770*