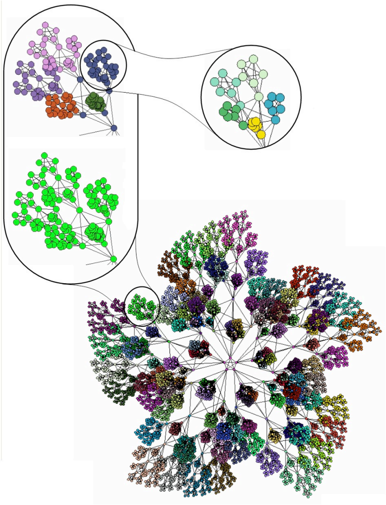
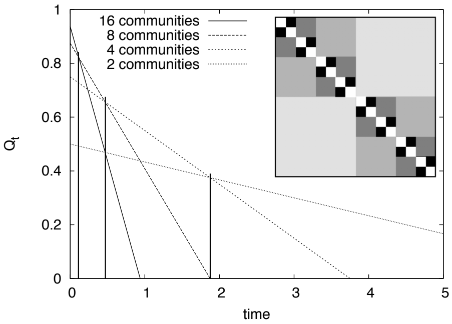
Finding communities at different resolutions in large networks

R. Lambiotte
Institute for Mathematical Sciences, Imperial College London,
53 Prince's Gate, South Kensington campus, SW7 2PG, UK



By tuning the resolution parameter *t*, it is possible to uncover communties of different characteristic sizes



On this simple hierarchical network of *16* nodes, whose adjacency matrix is represented in the upper right corner, the optimisation of Q_t leads to different partitions at different times. At small values of *t*, the optimal partition is made of *16* modules, but the this number decreases by plateau when *t* is increased.

In this short report, we present two methods allowing to uncover the optimal partitions of a network for the quantity

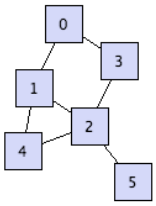
$$Q_t = (1-t) + \frac{1}{2m}\sum_{i,j}\left[tA_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i, c_j) \qquad (1)$$

introduced in [1,2]. In this definition, *N* is the total number of nodes in the network, *A* is the adjacency matrix of a symmetric network, i.e. A_{ij} is the weight of the link between node *i* and node *j*, k_i is the degree of node *i*, *m* is the total weight of the links (or total number of links for an unweighted network), c_i is the community of node *i* and *t* is a resolution parameter.

This quantity is equivalent, up to a multiplicative constant, to the tuneable Hamiltonian proposed by Reichardt and Bornholdt [2] (with the relation $t=1/\gamma$, γ being the resolution parameter of their method), and is equivalent to modularity when *t=1*. It is possible to show that the optimal partitions are made of larger and larger communities when *t* is increased. The optimal partitions of the limit cases *t=0* and *t=∞* are made of *N* single nodes and of one community containing the whole network respectively.

We have developed two computer programs in order to optimise this quality function and therefore to find communities at the desired resolution. Both programs take as an input the value of *t* and the name of a file storing the list of edges, e.g.

0 3
1 2
2 3
1 4
4 2
2 5
1 0



N.B. It is preferable (but not compulsory) that the node numbers are consecutive and start from *0*.

1- The first method generalises the Simulated Annealing (SA) method proposed by Guimera et al. [3] in order to optimise modularity. SA is known to give excellent results in terms of modularity optimisation but is limited to relatively small system sizes (say a few thousands of nodes). The program, called *netcarto*, is a modification of the original program written by

The codes are available at http://www.lambiotte.be

Here is the output of the SA algirithm when it is applied to the so-called football network [5] for values of *t* between *0* and *1*:

```
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 51
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
```

And the output of the Louvain algorithm for the same network and the same values of *t:*

```
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
```

One observes that the Louvain outperforms SA in terms of computation time, but that it even performs as well (and sometimes better) in terms of modularity optimisation.

N.B. The precision of the SA method could in principle be improved by changing the value of the cooling factor, but it would make it still slower. For instance, by setting *cooling factor=0.999*, one finds for *t=1.0*:

```
1.0 115 0.604429 10 186
```

which is better than previously (*0.601357*), but still not as good as the result found by the Louvain method.

R. Guimera and is written in C. A typical command line is

```
/usr/local/bin/netcarto 0.5 file.txt
```

where *t=0.5*, file.txt contains the network. The above line applies if netcarto has been installed in */usr/local/bin/*. As an output, the following line will be printed on the screen

t N $MaxQ_t$ N_c *seconds*

where *t* is the resolution parameter, *N* the number of nodes, $MaxQ_t$ the optimal value of Q_t found by the algorithm, N_c the number of communities in the optimal partition and *seconds* the number of seconds taken to find it. Information about the modules is appended to the file *modules.dat,* in the following format:

```
node0 community_of_node0
node1 community_of_node1
node2 community_of_node2
etc.
```

For an installation guide, we refer to the *README* file located in the folder *SA.*

N.B. The default values of the SA method have been set to: *iteration factor=1.0*; *cooling factor=0.995*. It is possible to change these values in the file *main_netcarto.c.*

2- The second method generalizes the so-called Louvain method for optimising modularity [4]. This greedy algorithm has been implemented in C++ and allows to study very large networks (the largest network studied so far was made of 10^8 nodes). It is a modification of the original program written by J.-L. Guillaume. Typical command lines are

```
./convert -i file.txt -o graph.bin
```

in order to convert the network file into the right format (into graph.bin) and then

```
./community graph.bin -l -1 -t 0.5 > temp.dat
./hierarchy temp.dat >> communities.dat
rm temp.dat
```

so that the outputs are the same as for the SA method. For an installation guide, we refer to the *readme.txt* file located in the folder *Louvain.*

Acknowledgments I would like to thank R. Guimera and J.-L. Guillaume for providing their C and C++ codes, as well as my collaborators J.-C. Delvenne and M. Barahona.

[1] J.-C. Delvenne, S. Yaliraki and M. Barahona, *arXiv...*

[2] R.Lambiotte, J.-C. Delvenne and M. Barahona, *arXiv:0812.1770*

[2] J. Reichardt and S. Bornholdt, *Phys. Rev. E* **74**, 016110 (2006)

[3] R. Guimera, M. Sales and L.A.N. Amaral, *Phys. Rev. E* **70**, 025101 (2004)

[4] V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, *J. Stat. Mech.*, P10008 (2008)

[5] M. Girvan and M. E. J. Newman, *Proc. Natl. Acad. Sci. USA* **99**, 7821-7826 (2002)

The codes are available at http://www.lambiotte.be