Relative Evaluation of Partition Algorithms for Complex Networks

Günce Keziban Orman, Vincent Labatut

Relative Evaluation of Partition Algorithms for Complex Networks

Günce Keziban Orman[1,2], Vincent Labatut[1]
[1]*Galatasaray University, Computer Science Department, stanbul, Turkey*
[2]*TUBITAK, Gebze/Kocaeli, Turkey*
keziban.orman@uekae.tubitak.gov.tr
vlabatut@gsu.edu.tr

Abstract

Complex networks partitioning consists in identifying denser groups of nodes. This popular research topic has applications in many fields such as biology, social sciences and physics. This led to many different partition algorithms, most of them based on Newman's modularity measure, which estimates the quality of a partition. Until now, these algorithms were tested only on a few real networks or unrealistic artificial ones. In this work, we use the more realistic generative model developed by Lancichinetti et al. to compare seven algorithms: Edge-betweenness, Eigenvector, Fast Greedy, Label Propagation, Markov Clustering, Spinglass and Walktrap. We used normalized mutual information (NMI) to assess their performances. Our results show Spinglass and Walktrap are above the others in terms of quality, while Markov Clustering and Edge-Betweenness also achieve good performance. Additionally, we compared NMI and modularity and observed they are not necessarily related: some algorithms produce better partitions while getting lower modularity.

1. Introduction

Complex networks have recently become very popular to model systems of interacting objects. They have been used in very different application domains, such as physics, biology, computer networks and social science [1].

Network partition, also called community detection, consists in looking for subsets of nodes densely interconnected relatively to the rest of the network [2]. It is one of the most active research areas in complex networks. Many different community detection algorithms have been proposed. They are generally based on classical hierarchical clustering principles adapted to graphs, or use optimization methods. Most of them use Newman's modularity measure [2], which estimates the quality of a given network partition relatively to the structure of the considered network.

Authors traditionally test their community detection algorithms on a few real networks [3-8]. But building such networks is a costly and difficult task, and determining reference communities is generally performed manually. This possibly results in small networks, where actual communities are not always defined objectively, or even known. Additionally, it is not possible to control the networks structural parameters such as degree distribution, transitivity or average distance. Consequently, the algorithm is tested on a very specific and limited set of features.

Artificial networks overcome these limitations, by allowing to control the networks structure, to define the communities *a priori* and to generate many networks randomly. This is the reason why many authors also use them to test their algorithms [2, 3, 7, 9, 10] But simple and not very realistic generative models such as Newman's [3] are used, hence the results cannot be trustfully transposed to real networks [11, 12]. Some recent works introduced more realistic models, able to mimic some of the real networks features. For instance, Lancichinetti *et al.* [12] proposed a model able to generate networks with controlled power-law degree and community size distributions.

In this work we used this model to generate a set of artificial networks with various sizes and properties, and applied seven different community detection algorithms on them. The normalized mutual information measure [13, 14] allowed us to assess the quality of the results and compare the considered algorithms. Additionally, we studied the behavior of the modularity measure in function of the algorithms performances.

In section 2, we describe the general mechanisms of community detection, define the modularity measure, and present the algorithms we chose to compare. In section 3, we explain how we generated our test set of artificial networks, and describe the normalized mutual information measure. In section 4, we present and discuss our results concerning the compared algorithms performances and modularity levels.

2. Community detection

2.1. General principle

Early solutions in community detection are based on hierarchical approaches whose result is a tree of communities called dendrogram. On the one hand, agglomerative approaches [7, 9, 15] start with as many communities as nodes, each node having its own community, and iteratively merge them until only one giant community remains. On the other hand, divisive approaches [3] start with one community containing all nodes, and iteratively split the communities until each node constitute one community. The merged or split communities are chosen accordingly to some distance or similarity function that allows detecting which communities are the most similar (agglomerative approach) or different (divisive approach). What distinguishes algorithms is mainly the nature of the distance or similarity function.

Some other approaches are optimization-based and use a heuristic function to estimate the quality of a network partition [16-18]. The general algorithm consists in first processing several partitions of the network (randomly or by following a fitting function) and second keeping the best one according to the estimated quality. This partition can then be refined in order to improve its quality.

The remaining approaches do not follow any common principle. For instance, some are inspired by other classical clustering approaches like density-based clustering [4]; some are agent-based [5]; some allows finding overlapping communities (one node can be a part of several communities at once) [19]; some use a latent space approach to process the probability for a node to belong to a community [6].

2.2. Modularity

Many community detection algorithms rely on Newman's modularity [2]. Some use it internally at each iteration, for instance as an optimization criterion. Some use it after the processing, for instance to select the best cut in a dendrogram built by a hierarchical approach. It is a costly measure to process, hence the numerous algorithms defined for its optimization [16].

Modularity was presented by Newman and Girvan [2] to assess the quality of a network partition. They first define what could be called a community contingency matrix, whose elements e_{ij} represent the total fraction of links from a node in community i towards a node in community j. We are considering undirected networks, so $e_{ij} = e_{ji}$ and this matrix is symmetric. The sum on row or column i is noted $a_i = \sum_j e_{ij}$ and represents the fraction of links with at least one node in community i.

The modularity measure is then defined as the difference between the observed and expected fractions of links inside a community, summed over all communities:

$$Q = \sum_i (e_{ii} - a_i^2)$$

The term e_{ii} corresponds to the observed fraction of links inside a community i, and a_i^2 is an estimation of this value under the hypothesis of uniformly randomly distributed links.

When the communities are not better than a random partition, or when the network does not exhibit any community structure, Q is negative or null. Its superior limit is 1, which can be approached only if the network has a strong community structure and if the communities have been perfectly detected [2].

2.3. Selected algorithms

Because we had many generated networks to process, we decided to focus first on fast and simple algorithms. When the result of the partition was a dendrogram, we used modularity to estimate its best cut.

Newman *et al.*'s Fast Greedy (FG) [7, 15] is modularity-based and relies on a hierarchical agglomerative approach. Its name is due to the use of a standard greedy method, making it relatively faster than earlier algorithms, and allowing it to process large networks.

Pons and Latapy's Walktrap (WT) [9] uses a hierarchical agglomerative method, where the distance between two nodes is defined in terms of random walk process. The random walks length must be specified through a parameter.

Girvan and Newman's Edge-betweenness (EB) [3] uses a hierarchical divisive method based on a link centrality measure. At each step it removes the most central links, creating a hierarchy of communities.

Newman's Eigenvector (EV) [16] is a modularity-based optimization method inspired by graph partitioning techniques. It relies on the eigenvectors of a so-called modularity matrix, instead of the graph Laplacian used in classic graph partitioning.

Van Dongen's Markov Clustering (MC) [18] also uses random walks, but this time to perform optimization through a bootstrapping procedure. An *inflation* parameter allows controlling the granularity of the resulting communities: the higher this parameter, the smaller the communities.

Reichardt and Bornholdt's Spinglass (SG) [17] is an optimization method relying on an analogy between the statistical mechanics of complex networks and physical spin models. A *spin* parameter represents the maximum number of communities the algorithm is allowed to find.

Raghavan *et al.*'s Label Propagation (PG) [8] uses the concept of node neighborhood and the diffusion of information in a network. Initially, each node is labeled with a unique value, representing its community. Then an iterative process takes place, where a node is given the label the most spread in its neighborhood.

3. Method

3.1. Network generation

In many community detection works [2, 3, 7, 10], artificial networks with a community structure are generated by using models comparable to Newman and Girvan's one [3, 7]. It relies on the principle of the Erdős-Rényi model [20], and produce networks with a degree following a Poisson distribution. Yet, it is well known that in most real networks, the degree follows a power-law distribution [1]. Networks with this property are called scale-free, because their degree distribution does not depend on their size (some other properties may, though). Moreover, in Newman and Girvan's approach, all the communities have the same size, whereas in real networks, community size is supposed to follow a power-law distribution too.

If we want our comparison to hold when the algorithms are applied on real networks, our artificial networks properties must be the most similar possible to those of real networks. For this reason, we chose to use a more recent generative model defined by Lancichinetti *et al.* [12]. It allows generating random networks with a community structure and a power-law degree distribution. Moreover, the size of the resulting communities also follows a power-law distribution.

The model needs the following compulsory parameters: the number of nodes n, the desired average $\langle k \rangle$ and maximum k_{max} degrees, the exponent γ for the degree distribution, the exponent β for the community size distribution, and a value μ called the mixing coefficient. The latter represents the average proportion of links between a node and other nodes located outside its community, $1-\mu$ consequently being the proportion of links with other nodes located in the same community.

The generation is performed in three steps. First, the well-known configuration model [21] is used to generate a scale-free network corresponding to the specified γ parameter. Second, the community sizes are drawn in accordance with the β parameter, and each node is randomly affected to a compatible community. Compatible means here that the community size must be greater or equal to the node internal degree. Some specific mechanisms ensure the convergence of the processing, see [12] for more details. Third, some links are rewired in order to respect the mixing coefficient. For a given node, the degree is not modified, but the ratio of internal and external links is changed so that the resulting proportion approximately respects μ.

For some of these parameters, we used values estimated from measurements on real networks. Experimental studies show that the γ coefficient usually ranges from 2 to 3 [1, 22]. The average and maximal degrees generally depend on the number of nodes in the network. For a scale-free network, it is estimated to be $\langle k \rangle \sim k_{max}^{-\gamma+2}$ [22] and $k_{max} \sim n^{1/(\gamma-1)}$ [1], respectively. The power-law parameter β for the community size distribution is known to range from 1 to 2 [15].

For n, the number of nodes, we could not afford using a realistic value because of computational constraints, so we limited the network size to 100 and 500 nodes. The parameter μ is of utmost importance, because it represents the community sharpness. For this reason, we generated networks with values in $[0.05; 0.95]$ with a 0.05 step.

For each set of parameters, we generated 10 networks in order to deal with possible discrepancies in the networks properties due to the random nature of the generation process. In rare occasions, we observed networks with a few disconnected components. Some algorithms like Walktrap cannot be applied on such networks, so we decided to connect them uniformly randomly.

3.2. Performance assessment

Another important point is the assessment of the results quality, which must be reliable in order to compare efficiently the communities detected by the tested algorithms.

As we stated before, the modularity measure is a standard for assessing the quality of a network partition. But it was designed to be used as an estimation function to guide community detection algorithms when the actual communities are unknown. The computed value for a given situation depends on both the quality of the detected communities and of the nature of the network community structure.

This dependence to the network structure prevents from using modularity to compare algorithm performances on different networks. Furthermore, we will use artificial networks, whose communities are known *a priori*. In this context, modularity is not an appropriate measure, because it does not make use of this important information. For interpretation purposes, we nevertheless processed modularity for each partition.

In place of the modularity, we used the normalized mutual information measure (NMI). It has been used in the context of classical clustering to compare two different partitions of a data set [13]. It was shown to be an efficient way to assess the

quality of estimated network communities by Danon *et al.* [14].

The measure is derived from a confusion matrix whose element m_{ij} represents the number of nodes put in community i by the considered algorithm, when they actually belong to community j. This matrix is usually rectangular, because the algorithm does not necessarily find the correct number of communities.

$$I = \frac{-2\sum_i \sum_j m_{ij} \log\left(n m_{ij} / m_{i+} m_{+j}\right)}{\sum_i m_{i+} \log(m_{i+}/n) + \sum_j m_{+j} \log\left(m_{+j}/n\right)}$$

The terms $m_{i+}$and m_{+j} represent the sums of the row i and column j elements, respectively. If the estimated communities correspond perfectly to reality, the measure takes the value 1, whereas it is 0 when the estimated communities are independent from the actual ones.

4. Results and discussion

We analyzed the algorithms performances in function of the parameters used to generate the networks. The results were very similar for $n = 100$ and $n = 500$, so we will just discuss the latter here. It was found that β and γ do not seem to significantly affect any of the algorithms: their (Pearson's) correlation to the results is always less than 0.06. On the contrary, μ has a strong effect (correlations below -0.5) and the average and maximum degrees exhibit non-trivial correlations (above 0.35).

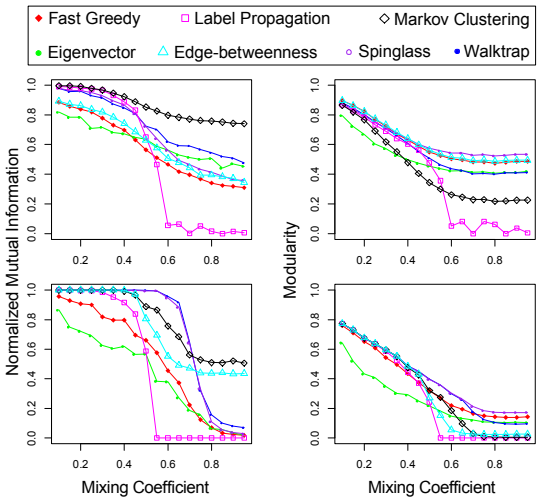

Fig. 1. Comparison of the seven algorithms results for $n = 500$, $\beta = 2$ and $\gamma = 3$. On the first row, the average and maximum degrees are $\langle k \rangle = 5$ and $k_{max} = 15$, and on the second row they are 30 and 90, respectively. Performance is expressed in terms of normalized mutual information on the left column, and modularity on the right one, in function of μ. Each point corresponds to an average over 10 generated networks.

The left part of Fig. 1 shows the results for all seven algorithms, expressed in terms of NMI in function of μ. The plots take the form of reversed sigmoid functions. In other terms, for all algorithms, the performances stay high when μ is small, then start to decrease when μ reaches a certain value corresponding to an inflection point, and finally stabilize again at a lowest value. Indeed, an increase in μ makes the community structure vanish (the proportion of links directed outside the community gets bigger), and thus harder to be identified. Performances are close, but some differences exist however: the slope and the location of the inflection point vary according to the considered algorithm. FG and EV performances decrease almost linearly with μ, with an inflection point around $\mu = 0.5$. LP has the strongest slope, with an inflection point also around $\mu = 0.5$. For WT and SG, the slope is gentler and the inflection point is shifted towards $\mu = 0.7$. Finally, MC and EB have an even gentler slope, with an inflection point around $0.5; 0.6$, and their minimal performances stay significantly above other algorithms results after the inflection point. Dispersion is not shown on the graph, because EV and LP performances can vary a lot (especially during the performance drop), making the plots difficult to read. Each point represents an average over 10 results, and this may not be enough to obtain stable values. The other algorithms do show very small dispersion, though.

The right part of Fig. 1 shows the same results, but this time assessed through Newman's modularity. For each algorithm, we observe an almost linear decrease until μ reaches $0.5; 0.6$, and then stabilization around a lowest value. An interesting question is whether the relative order of the algorithms is the same when we consider NMI and modularity values. Indeed, modularity is known to have some flaws, for example it is sensitive to community size [11] and it is possible to find partitions of Poisson random networks with relatively high modularity values, even though they have no community structure [23]. A preserved order relatively to NMI would show that modularity constitutes a good estimation of the partition quality, since NMI uses the actual partition to assess the algorithms performances. When μ is below 0.5, we observe roughly the same relative order between NMI and modularity. There are a few exceptions though, like with $\langle k \rangle = 5$, $\gamma = 3$, $\beta = 2$: for a μ close to 0.5, MC is first for the NMI value but penultimate for the modularity. When μ is above 0.5, the order is not preserved at all. In the same example, when μ gets close to 1, the relative positions are different not only for MC, but for almost every algorithm.

Beside μ, the maximum and average degrees also have a significant effect on the observed performances. As Lancichinetti *et al.* also noticed for two other algorithms [12], a higher degree makes the performances better. This is actually true only when μ is small enough. But when μ is above $0.5; 0.6$, an

increase in the degree causes the performances to drop rapidly.

The 0.5 value is very important, because when μ is above this threshold, each node has more links outside its community than inside. In other terms, the communities cannot be described anymore as "subsets of nodes densely interconnected relatively to the rest of the network". All the tested algorithms rely on this definition of the community concept, so it makes sense to observe a performance drop when it does not hold anymore. This is also true for the modularity measure, which relies on the same definition to estimate a partition quality. This explain the clear differences noticed between modularity and NMI when $\mu > 0.5$. It also explains why a degree increase has a positive effect when $\mu < 0.5$, whereas it is negative otherwise. Indeed, below 0.5, each link represents pertinent information in terms of discovering the community structure. A higher degree means more information, hence better performances. On the contrary, when $\mu > 0.5$, the links do not correspond to the community structure, therefore they are noise. A higher degree means more noise, making the community structure harder to find, and leading to lower performances.

What is extremely surprising in the presented results is EB and MC good performances for very high μ values. For instance, MC almost reaches a 0.8 NMI for $\mu = 0.95$ ($\langle k \rangle = 5$, $\beta = 2$, $\gamma = 3$), which means it can nearly find the exact communities when the community structure does not actually exist. The only hypothesis we can advance to explain this observation is that NMI seems to be sensitive to the number of estimated communities, which may unrealistically increase the measured performances for algorithms tending to create many communities.

Since the definition of community holds only when $\mu \leq 0.5$, it makes no sense to compare the algorithms for higher values. Under these conditions, the criterion to select the best algorithm is to have the highest values on the [0,0.5] interval, which translate in a high initial value (i.e. for a small μ), a strong slope and an inflection point located far on the right (high μ). For a low average degree, MC has the best results, because it stays on top even when the communities are not clearly defined (μ close to 0.5). LP, SP and WT are very close to MC; and EB, EV and FG performances are clearly below. When the average degree is higher, it is more difficult to discriminate the algorithms. EB, MC, SG and WT are excellent (close to the maximum until $\mu = 0.5$), with an advantage to SG and WT, whose performances start declining later. FG and LP have almost the same performances than for the lower degree, and they even decrease for EV, which means they are below the other algorithms.

5. Conclusion

In this paper, we compared seven different community detection algorithms, namely: Edge-betweenness (EB) [3], Eigenvector (EV) [16], Fast Greedy (FG) [7, 15], Label Propagation (LP) [8], Markov Clustering (MC) [18], Spinglass (SP) [17] and Walktrap (WT) [9]. We used a set of artificial networks generated with the model defined by Lancichinetti *et al.* [12], which allows randomly producing networks with a community structure and power-law degree and community size distributions. This model uses a μ parameter to control the communities sharpness, and we clearly stated that comparing algorithms was relevent only for $\mu \leq 0.5$. We used the normalized mutual information measure [13, 14] to assess the algorithms performances.

For a low average degree, all algorithms perform relatively well, with MC clearly standing out, and EB, EV and FG performing below the other algorithms. For a higher average degree, EV and FG clearly have poor results relatively to the other algorithms, whose performances are so excellent it is difficult to discriminate them, especially EB, MC, SG and WT. WT and SG performs slightly better though, because their performance drop starts after the 0.5 μ threshold. To expose more important differences amongst the algorithms performances, it may be necessary to perform some further experiments with larger networks. Moreover, other factors than performance are also important for algorithms comparison, like for instance memory and time complexities. Here, for a comparable partition quality, WT and MC are much faster than EB and SG.

Of course, there are some limitations to our results. The generative model we used is more realistic than earlier ones, but it is based on the configuration model [21], which is known to produce networks with no degree correlation and whose transitivity tends to zero for some parameters [1]. On the contrary, real networks show non-null degree correlation (positive or negative), and their transitivity is high [1]. Maybe this could be corrected by using another model instead, such as preferential attachment [24] (or one of its variations), able to generate networks with more realistic properties. Of course there is no certainty about whether or not these properties would resist the modifications performed on the initial network by Lancichinetti *et al*'s. [12] generation process, when the community structure is created. Moreover, real networks properties are usually described commonly, but there may be strong differences between the various types of real networks such as social networks, biological networks, information networks, etc. [1]. In that case, a proper test should compare algorithms on different types of corresponding artificial networks.

We compared the algorithms on networks containing 100 and 500 nodes only. Real networks are generally much bigger, in the order of tens of thousands or millions of nodes. For more significance, the algorithms should be tested on this type of networks, but this raises two problems: 1) processing community detection on such huge networks is significantly more time expensive, and 2) determining realistic values for the average and maximal degrees is difficult because of the heterogeneity observed in real networks for these properties. The second point is important, since we observed the performances vary strongly in function of these sole properties. We also limited this comparison to the fastest algorithms, again for computability and time reasons. A proper exhaustive test should consider more expensive algorithms (see [14]).

6. References

[1] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review,* vol. 45, pp. 167-256, 2003.

[2] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E,* vol. 69, p. 026113, 2004.

[3] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 99, pp. 7821-7826, Jun 11 2002.

[4] T. Falkowski, A. Barth, and M. Spiliopoulou, "DENGRAPH: A Density-based Community Detection Algorithm," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2007, pp. 112-115.

[5] Y. Liu, Q. Wang, Q. Wang, Q. Yao, and Y. Liu, "Email Community Detection Using Artificial Ant Colony Clustering," in *Advances in Web and Network Technologies, and Information Management* Berlin / Heidelberg: Springer, 2007, pp. 287-298.

[6] P. Hoff, A. Raftery, and M. Handcock, "Latent space approaches to social network analysis," *Journal of the American Statistical Association,* vol. 97, pp. 1090-1098, 2002.

[7] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E,* vol. 69, p. 066133, Jun 2004.

[8] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E,* vol. 76, p. 036106, 2007.

[9] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *Computer and Information Sciences - Iscis 2005, Proceedings,* vol. 3733, pp. 284-293, 2005.

[10] J. Reichardt and S. Bornholdt, "Detecting Fuzzy Community Structures in Complex Networks with a Potts Model," *Physical Review Letters,* vol. 93, p. 218701, 2004.

[11] L. Danon, A. Diaz-Guilera, and A. Arenas, "The effect of size heterogeneity on community identification in complex networks," *Journal of Statistical Mechanics-Theory and Experiment,* p. 11010, Nov 2006.

[12] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys Rev E Stat Nonlin Soft Matter Phys,* vol. 78, p. 046110, Oct 2008.

[13] A. L. N. Fred and A. K. Jain, "Robust Data Clustering," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. vol. 2: IEEE Computer Society, 2003, pp. 128-136.

[14] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *J. Stat. Mech.,* p. P09008, 2005.

[15] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E,* vol. 70, p. 066111, 2004.

[16] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E,* vol. 74, p. 036104, 2006.

[17] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Phys. Rev. E,* vol. 74, p. 016110, 2006.

[18] S. van Dongen, "Graph clustering by flow simulation." vol. PhD Utrecht: University of Utrecht, Netherlands, 2000.

[19] I. Derenyi, G. Palla, and T. Vicsek, "Clique percolation in random networks," *Physical Review Letters,* vol. 94, Apr 29 2005.

[20] P. Erdos and A. Renyi, "On random graphs," *Publicationes Mathematicae,* vol. 6, pp. 290-297, 1959.

[21] M. Molloy and B. Reed, "A critical point for random graphs with a given degree sequence," *Random Structures and Algorithms,* vol. 6, pp. 161-179, 1995.

[22] A. Barabasi and R. Albert, "Statistical mechanics of complex networks," *Reviews of Modern physics,* vol. 74, pp. 47-96, 2002.

[23] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral, "Modularity from fluctuations in random graphs and complex networks," *Physical Review E,* vol. 70, p. 025101, 2004.

[24] A.-L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science,* vol. 286, pp. 509-512, 1999.