Directed Louvain : maximizing modularity in directed networks

Research · November 2015

DOI: 10.13140/RG.2.1.4497.0328

2 authors:

Nicolas Dugué
Université du Maine
31 PUBLICATIONS **233** CITATIONS
SEE PROFILE

Anthony Perez
Université d'Orléans
45 PUBLICATIONS **342** CITATIONS
SEE PROFILE

Directed Louvain : maximizing modularity in directed networks

Nicolas Dugué Anthony Perez

November 20, 2015

Abstract

In this paper we consider the community detection problem from two different perspectives. We first want to be able to compute communities for *large directed networks*, containing *million* vertices and *billion* arcs. Moreover, in a large number of applications, the graphs modelizing such networks are *directed*. Nevertheless, one is often forced to forget the direction between the connections, either for the sake of simplicity or because no other options are available. This is in particular the case on large networks, since there are only a few scalable algorithms at the time. We thus turn our attention to one of the most famous scalable algorithms, namely Louvain's algorithm [3], based on modularity maximization. We modify Louvain's algorithm to handle directed networks based on the notion of directed modularity defined by Leicht and Newman [13], and provide an empirical and theoretical study to show that one should prefer directed modularity. To illustrate this fact, we use the LFR benchmarks by Lancichinetti and Fortunato [8] to design an evaluation benchmark of directed graphs with community structure. We also give some examples and insights on the situations where one should *really* consider direction when maximizing modularity. Finally, for the sake of completeness, we compare the results obtained with Oslom [12], one of the best algorithms to detect communities in directed networks. While the results obtained with such an algorithm are by far better on the LFR benchmarks, we emphasize that it is still not well-suited to deal with very large networks.

1 Introduction

In various domains such as social networks or bioinformatics, being able to detect communities efficiently constitutes a very important research interest [6]. In most cases, the underlying graphs representing data are *directed*. This happens for instance when considering some social network graphs, where relations between two users u and v can be represented by stating that u has an influence *over* v rather than simply saying that they both interact. It thus seems quite obvious to consider direction when detecting communities, and several algorithms were proposed in this sense, such as Oslom [12] or InfoMap [19, 20]. In this article, we are interested in detecting communities in very large networks such as the Twitter graph [4], which contains more than 50 *millions* vertices and almost 2 *billions* arcs. As we shall see Section 5, Oslom [12] fail to efficiently produce communities when considering such networks [4], especially if one wants to use only a few computer ressources. Due to this fact, a common solution is to simply *forget* direction when detecting communities in really large network and to run Louvain's algorithm [3] (which is extremely well-suited for large networks). To the best of our knowledge, there is no version of this algorithm maximizing *directed modularity* [13]. This fact can also be seen in a survey comparing algorithms for community detection, where Lancichinetti and Fortunato [10] did not even consider Louvain's

algorithm for their directed networks analysis. Instead, they used simulated annealing for modularity optimization [6] even if they confirmed on undirected networks that Louvain's algorithm performs better and runs a lot faster.

Our results. In this work, we give some insights about the importance of direction while detecting communities. To that aim, we consider Louvain's algorithm [3], which is implemented for non-directed graphs *only*. By modifying the existing source code [2], we manage to deal with directed graphs, following the notion of directed modularity introduced by Leicht and Newman [13] (Section 2). We then generated a benchmark of directed graphs using the framework provided by Fortunato et al. [8], and computed communities on such graphs using both versions of the Louvain's algorithm[1]. Our results show strong evidence that direction is important when detecting communities (Section 5). Finally, we also compare these results to communities obtained by a recent community detection algorithm called OSLOM [12], both from the semantic and complexity viewpoints (Section 5). We emphasize that OSLOM [12] cannot deal with large graphs such as the Twitter graph [4] (billions of edges), while Louvain's algorithm produces results in a couple of hours.

2 Detecting community in large (directed) networks

Modularity. A classic way of detecting communities is to find a partition of the vertex set that maximizes an optimization function. One of the most famous optimization function is called *modularity* [16]. This function provides a way to value the existence of an edge between two vertices of an undirected network by comparing it with the probability of having such an edge in a random model following the same degree distribution than the original network. For instance, an edge between two vertices of large degree is not surprising, and thus does not contribute much to the modularity of a given partition, whereas an edge between two vertices of small degree is more surprising. Formally, the modularity Q of a partition $\mathcal{C}$ of an undirected graph $G = (V, E)$ is defined as follows :

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

where m stands for the number of edges of G, A_{ij} represents the weight of the edge between i and j (set to 0 if such an edge does not exist), d_i is the degree of vertex i (i.e. the number of neighbors of i), c_i is the community to which vertex i belongs and the δ-function $\delta(u, v)$ is defined as 1 if $u = v$, and 0 otherwise.

Leicht and Newman [13] adapted the notion of modularity for directed graphs, motivated by the following observation: if two vertices u and v have small in-degree/large out-degree and small out-degree/large in-degree, then having an arc from v to u should be considered more surprising than having an arc from u to v. Taking this into account, the definition for directed modularity of a partition of a directed network can be easily formulated:

[1] Louvain's algorithm is usually non-deterministic, but in order to obtain consistent results, we *always* consider the vertices in the same order.

$$Q_d = \frac{1}{m} \sum_{i,j} \left[A_{ij} - \frac{d_i^{in} d_j^{out}}{m} \right] \delta(c_i, c_j)$$

where A_{ij} now represents the existence of an *arc* between i and j and d_i^{in} (resp. d_j^{out}) stands for the *in-degree* (resp. *out-degree*) of i.

Louvain's algorithm. We now briefly describe the behavior of Louvain's algorithm to maximize modularity. The algorithm is the same for both the classic and directed versions of modularity. It relies on a greedy procedure: starting from any partition of the vertices (usually the partition into singletons), the algorithm tries to increase the value of modularity by moving vertices from their community to any other neighbor one. In other words, the algorithm computes the *gain* of modularity obtained by adding vertex i to community C as follows (for the undirected case):

$$\begin{aligned} \Delta_Q &= \left[\frac{\sum_{in} + d_i^C}{2m} - \left(\frac{\sum_{tot} + d_i}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{d_i}{2m} \right)^2 \right] \\ &= \frac{d_i^C}{2m} - \frac{\sum_{tot} \cdot d_i}{2m^2} \end{aligned}$$

where d_i^C denotes the degree of node i in community C, $\sum_{in}$ the number of edges contained in community C and $\sum_{tot}$ the total number of edges incident to community C. Actually, the first formula is the one as described in [3], but one can see that it reduces to the second one. The algorithm does a similar calculation to compute the *gain* obtained by *removing* vertex i from its own community C_i in a first place. The algorithm carries on as long as it exists a move that improves the value of modularity.

The behavior of the algorithm is *exactly* the same in the directed case, the main difference lying in the calculation for the gain of modularity obtained by adding vertex i to community C, which can now be done using the following:

$$\Delta_{Q_d} = \frac{d_i^C}{m} - \left[\frac{d_i^{out} \cdot \sum_{tot}^{in} + d_i^{in} \cdot \sum_{tot}^{out}}{m^2} \right]$$

where $\sum_{tot}^{in}$ (resp. $\sum_{tot}^{out}$) denotes the number of *in-going* (resp. *out-going*) arcs incident to community C.

3 Theoretical comparison between undirected and directed modularity

Our goal is to validate the observation made by Leicht and Newman [13] by showing what happens if one uses the modularity Q [16] on a directed graph instead of using its directed version Q_d [13]. To that aim, we use a straightforward case study where we consider two subgraphs C_1 and C_2 which both are communities of a directed network (see Figure 1). We are basically studying when merging these communities lead to a value increase of both modularities. To calculate the undirected modularity on a network which is usually directed, we have to ignore the links direction.

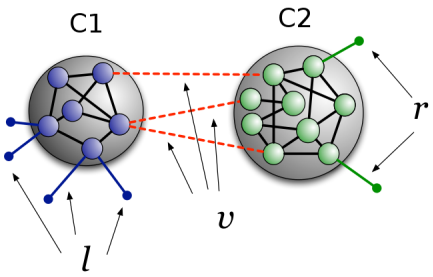

Figure 1: Figure extracted from the article of Lancichinetti and Fortunato [11].

Thus, if we are processing a directed network $D = (V, A)$ where $(u, v) \vee (v, u) \in A$, then $(u, v) \in E$ in the undirected version $G = (V, E)$. We use $Q^{C_1 \setminus C_2}$ (resp. $Q_d^{C_1 \setminus C_2}$) to refer to the undirected (resp. directed) modularity value of the network with C_1 and C_2 distinct communities. In the same way, we use $Q^{C_1 \cup C_2}$ (resp. $Q_d^{C_1 \cup C_2}$) to refer to the modularity value of the network where C_1 and C_2 are part of the same community. We name $A_{1,2}$ arcs between communities C_1 and C_2, and $E_{1,2}$ the corresponding edges in the undirected network. Considering the undirected case, $|E_{1,2}| = |A_{1,2}|$ if $\forall (u, v) \in A_{1,2}$ then $(v, u) \notin A_{1,2}$. We also have that $|E_{1,2}| = \frac{1}{2} \cdot |A_{1,2}|$ if $\forall (u, v) \in A_{1,2}$ then $(v, u) \in A_{1,2}$. Thus, $|E_{1,2}| \leqslant |A_{1,2}| \leqslant 2 \cdot |E_{1,2}|$.

3.1 Undirected case

When C_1 and C_2 are considered as part of the same community, $E_{1,2}$ links contribute to increase modularity value, as shown in bold in the following formula.

$$Q^{C_1 \cup C_2} = \left(\frac{d_{C_1}^{int}}{m} + \frac{d_{C_2}^{int}}{m} + \frac{\mathbf{|E_{1,2}|}}{\mathbf{m}} \right) - \left(\sum_{i,j \in C_1} \frac{d_i d_j}{4m^2} + \sum_{i,j \in C_2} \frac{d_i d_j}{4m^2} + \sum_{\mathbf{i \in C_1, j \in C_2}} \frac{\mathbf{d_i d_j}}{\mathbf{2m^2}} \right)$$

When C_1 and C_2 are splitted in two different communities, both the terms in bold before disappear.

$$Q^{C_1 \setminus C_2} = \left(\frac{d_{C_1}^{int}}{m} + \frac{d_{C_2}^{int}}{m} \right) - \left(\sum_{i,j \in C_1} \frac{d_i d_j}{4m^2} + \sum_{i,j \in C_2} \frac{d_i d_j}{4m^2} \right)$$

Thus, if summing these bold terms results in a positive number, C_1 and C_2 are merged. At the contrary, if the sum is negative, C_1 and C_2 are considered as two distinct communities. Therefore, studying when these communities are merged or not consists in studying the sum of these terms as follows.

$$\begin{aligned} \delta_Q &= Q^{C_1 \cup C_2} - Q^{C_1 \setminus C_2} \\ &= \frac{|E_{1,2}|}{m} - \sum_{i \in C_1, j \in C_2} \frac{d_i d_j}{2m^2} \\ &= \frac{1}{m} \Big(|E_{1,2}| - \sum_{i \in C_1, j \in C_2} \frac{d_i d_j}{2m} \Big) \end{aligned}$$

Hence:

$$\delta_Q > 0 \quad \Leftrightarrow \quad |E_{1,2}| > \sum_{i \in C_1, j \in C_2} \frac{d_i d_j}{2m} \tag{1}$$

3.2 Directed case

In the directed case, we obtain a quite similar result. Indeed, when C_1 and C_2 are considered as being part of the same community, we obtain:

$$\begin{aligned} \delta_{Q_d} &= Q_d^{C_1 \cup C_2} - Q_d^{C_1 \setminus C_2} \\ &= \frac{|A_{1,2}|}{2m} - \sum_{i \in C_1, j \in C_2} \frac{d_i^{in} d_j^{out}}{4m^2} - \sum_{i \in C_1, j \in C_2} \frac{d_i^{out} d_j^{in}}{4m^2} \end{aligned}$$

Hence:

$$\delta_{Q_d} > 0 <=> |A_{1,2}| > \sum_{i \in C_1, j \in C_2} \left(\frac{d_i^{in} d_j^{out}}{2m} + \frac{d_i^{out} d_j^{in}}{2m} \right)$$

3.3 Comparison

To compare the choices made by both modularities, we replace the vertex degree of Equation 1 by its in- and out-going counterparts.

$$d_i d_j = (d_i^{in} + d_i^{out})(d_j^{in} + d_j^{out})$$

We thus obtain the following equivalence :

$$|E_{1,2}| > \sum_{i \in C_1, j \in C_2} \left(\frac{d_i^{in} d_j^{out}}{2m} + \frac{d_i^{out} d_j^{in}}{2m} \right) + \sum_{i \in C_1, j \in C_2} \left(\frac{d_i^{in} d_j^{in}}{2m} + \frac{d_i^{out} d_j^{out}}{2m} \right)$$

Let us define the following terms :

$$S = \sum_{i \in C_1, j \in C_2} \left(\frac{d_i^{in} d_j^{out}}{2m} + \frac{d_i^{out} d_j^{in}}{2m} \right)$$

$$T = \sum_{i \in C_1, j \in C_2} \left(\frac{d_i^{in} d_j^{in}}{2m} + \frac{d_i^{out} d_j^{out}}{2m} \right)$$

Thus, in the undirected case, C_1 and C_2 are merged when $|E_{1,2}| > S + T$ while in the directed case, the fusion is done when $|A_{1,2}| > S$, T being absent from the equation.

The term S confirms the observation made by Leicht and Newman [13]. Furthermore, we can see that T is not relevant at all. Multiplying the incoming degrees in one side and the outgoing degrees in the other side does not allow to estimate links probability to exist between communities in a random network. This may explain the better results obtained with the Louvain algorithm implementing the directed modularity.

4 Analysis of the differences

We now study the conditions that make differences arise between the two versions of Louvain's algorithms. We first give some intuition on the configurations that can lead to two different moves in the algorithm, and then provide several examples where the difference is significant.

Sufficient conditions to have a difference. To complete the previous observations, we give some conditions that will influence community detection between the two methods. Recall that the *gain of modularity* can be easily computed (in both cases) using the following:

$$\begin{aligned} \Delta_Q &\sim d_i^C - \frac{\sum_{tot} \cdot d_i}{2m} \\ \Delta_{Q_d} &= d_i^C - \left[\frac{d_i^{out} \cdot \sum_{tot}^{in} + d_i^{in} \cdot \sum_{tot}^{out}}{m} \right] \end{aligned}$$

We thus have to study the behavior of the terms $\frac{\sum_{tot} \cdot d_i}{m}$ and $\frac{d_i^{out} \cdot \sum_{tot}^{in} + d_i^{in} \cdot \sum_{tot}^{out}}{m}$ for a given vertex i and a given community C. In particular, we want to express the conditions when the first one is positive and the second one negative, or vice-versa. Recall that, in the first case, the classic Louvain's algorithm will not consider adding vertex i to community C to increase modularity, while the directed version will do so.

Cases that make a difference. We first provide some simple examples when the classic Louvain's algorithm fails at detecting communities, whereas the algorithm maximizing directed modularity finds a perfect match with the ground truth communities.

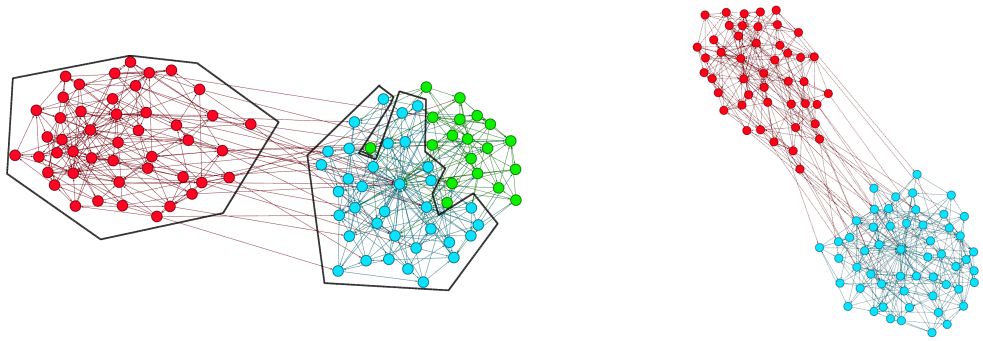

Figure 2: On the left, the three communities obtained by maximizing standard modularity are represented. On the right, the ones obtained using directed modularity.

Consider the graph represented Figure 2, which contains 100 vertices and 2 communities. When maximizing directed modularity, Louvain's algorithm succeeds in retrieving the communities whereas the classic modularity maximization fails to merge two communities. The explanation for this situation follows from our previous arguments. Indeed, the graph contains vertices with unbalanced in and out-degrees, who thus influence the greedy method of Louvain's algorithms. Such a situation can also be observed on larger graphs[2] (see Figure 3).

[2]For the sake of visiblity we do not consider larger graphs, but mention that similar situations happen also.

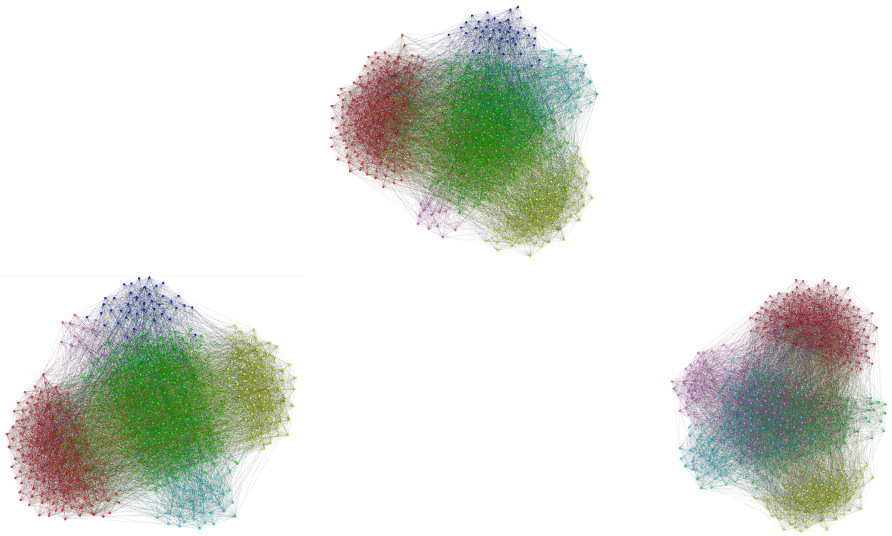

Figure 3: On top, the groundtruth communities. On the bottom left, the communities found by the directed version of the algorithm and on the bottom right the ones provided by the classic one.

5 Experimental results

We now present empirically the differences that arise between classic modularity maximization and the directed one. To that aim, we evaluate the results of both the modulairies over the so-called directed LFR benchmarks [10].We consider partitions (that is non-overlapping communities).

5.1 The LFR benchmarks

To validate the efficiency of Louvain algorithm adapted to directed graphs, we use benchmarks introduced by Lancichinetti and Fortunato [9]. These benchmarks allow to test community detection algorithms on directed graphs, and are designed in order to be as realistic as possible with respect to real networks. It is indeed possible to set important features such as the power-law distribution of the degrees of the nodes or of the communities sizes, as well as the maximum and average degrees of nodes in the graphs. Another major feaure introduced in these benchmarks is the mixing parameter. The mixing parameter allows to create graphs with communities more or less well-defined. A low mixing parameter indicates communities well-defined, and hence easy to detect. Reciprocally, a high mixing parameter allows to create graphs with communities which will be hard to detect.

5.2 Measures

To compare the results obtained by the community detection methods, we use three evaluation measures. The results of the community detection algorithms are thus compared with the communities defined by the benchmarks we use. In the following, we use *clustering* to denote the community sets obtained by the algorithms used. The term *cluster* is thus one community of these sets. We use *community* to talk about the groundtruth communities established by the benchmark.

The first measure, called *V-Measure* [18] is made of two criteria: *homogeneity* and *completeness*. This may be compared to the *F-measure* based on precision and recall measures. A clustering maximizes the homogeneity if for each cluster, we find only elements of a same community. Symetrically, completeness is maximized when for each community, all elements of a same community are in a single cluster. By computing the harmonic mean of these two values, we obtain the so-called *V-measure*. The second one is the NMI [21] for *Normalized Mutual Information*. Built upon concepts from information theory, this measure is commonly used to compare clusterings. Roughly speaking, this measure defines how much knowing one of two clusterings reduces uncertainty about the other. Thus, the higher the NMI, the more information the two clusterings share. We use the normalization introduced by Strehl and Gosh [21] defined as follows.

definition 1 (NMI [21]) *Let $\mathcal{U}$ and $\mathcal{V}$ be two clusterings. Then the* Normalized Mmutual Information *is defined as a function of the* mutual information I *and the* conditional entropy H*:*

$$NMI(\mathcal{U},\mathcal{V}) = \frac{I(\mathcal{U},\mathcal{V})}{\sqrt{H(\mathcal{U}) \cdot H(\mathcal{V})}}$$

Finally, to compute the Purity [24], we assign each cluster to the community which nodes are more frequent in the cluster. Then, by summing all well-classified nodes for each of these clusters and dividing it by the number of vertices, we obtain the accuracy of our clustering.

5.3 Classic LFR benchmarks

We begin this section by giving some results obtained by generating LFR benchmarks using parameters described by Lancichinetti and Fortunato [10]. Those parameters consider two different cases, namely graphs having *small* and *big* community sizes. In the first case, the communities are set to contain between 10 and 50 vertices, while they are required to contain between 20 and 100 vertices in the latter one. In such graphs, the average degree is set to 20 and the maximum degree is set to 50. We consider graphs having respectively 1000 and 5000 nodes, and make the mixing parameter go from 0.1 (*i.e.* well-defined communities) to 0.6. Finally, we set the power law distribution to 2 in all cases. We first give a general picture of the results we obtained w.r.t. to the number of vertices and the size of the communities (Table 1).

n	$minc$	$maxc$	# graphs	$\geqslant$	$\geqslant^{+0.05}$	$\geqslant^{+0.1}$	$\geqslant^{+0.2}$
1000	10	50	900	839	83	54	5
1000	20	100	900	776	70	27	3
5000	10	50	900	804	0	0	0
5000	20	100	900	785	54	31	3

Table 1: Proportion of graphs where the NMI of the classic modularity (nmi_o) is greater than the one of the directed modularity (nmi_d) by a given percentage.

Louvain's algorithm maximizing *directed* modularity is better in almost 75% of the cases. Moreover, we would like to mention that if the improvement is rather low on average, there are some interesting cases where the improvement is drastic.

We then compare the outputs of both version of Louvain's algorithms according to the aforementioned quality measures. We conducted such an experimentation by considering different networks sizes, and also by modifying the *mixing parameter*. As one can observe Table 2, the directed version (which corresponds to the bottom table) is better in most cases.

n	mu	NMI	V-measure	Homogeneity	Completeness	Purity
1000	0.1	0.987	0.987	1.000	0.975	1.000
1000	0.6	0.965	0.964	0.999	0.932	0.999
5000	0.1	0.966	0.965	1.000	0.934	1.000
5000	0.6	0.909	0.905	0.999	0.828	0.999

n	mu	NMI	V-measure	Homogeneity	Completeness	Purity
1000	0.1	**0.995**	**0.995**	1.000	**0.990**	1.000
1000	0.6	**0.978**	**0.978**	**1.000**	**0.958**	**1.000**
5000	0.1	**0.978**	**0.978**	**1.000**	**0.957**	1.000
5000	0.6	**0.920**	**0.917**	0.999	**0.848**	0.999

Table 2: Results obtained on the classic LFR benchmarks with the classic and the directed versions of Louvain algorithm. Each mesure indicates the average taken over 100 graphs with the indicated parameters.

5.4 Generated LFR benchmarks

In this Section, we present similar observations on a new set of benchmarks that we generated for this purpose. Recall that the average and maximum degree are respectively fixed to 20 and 50 in the classic LFR benchmarks. This seems quite unrealistic when trying to simulate social networks: on Table 3 we observe in several well-known complex networks that the average degree is in general much lower, while the maximum degree is much higher. Hence, it seems quite restrictive to impose a maximum degree of 50, when it is really common for vertices of such networks to have a degree close to $\frac{n}{3}$.

Network	Nodes	Edges	Avg degree	Max degree	Power-law
Zachary karate club [22]	34	78	4.59	17	2.16
Openflight [17]	2,939	30,501	20.76	473	1.79
Arxiv Astro-ph [14]	18,771	198,050	21.10	504	2.60
Internet AS [23]	34,761	171,403	9.86	5,305	1.86
SlashDot [5]	51,083	140,778	5.51	3,357	1.91
Flickr [15]	2,302,925	33,140,018	28.78	34,174	2.02
DBPedia [1]	2,152,642	7,494,124	6.96	329,714	2.15

Table 3: Basic properties of several physical, social or reference based network from literature.

To complete our analysis, we generated about 40000 graphs with different parameters closely related to those observed in real networks. In particular, we infered the average and maximum

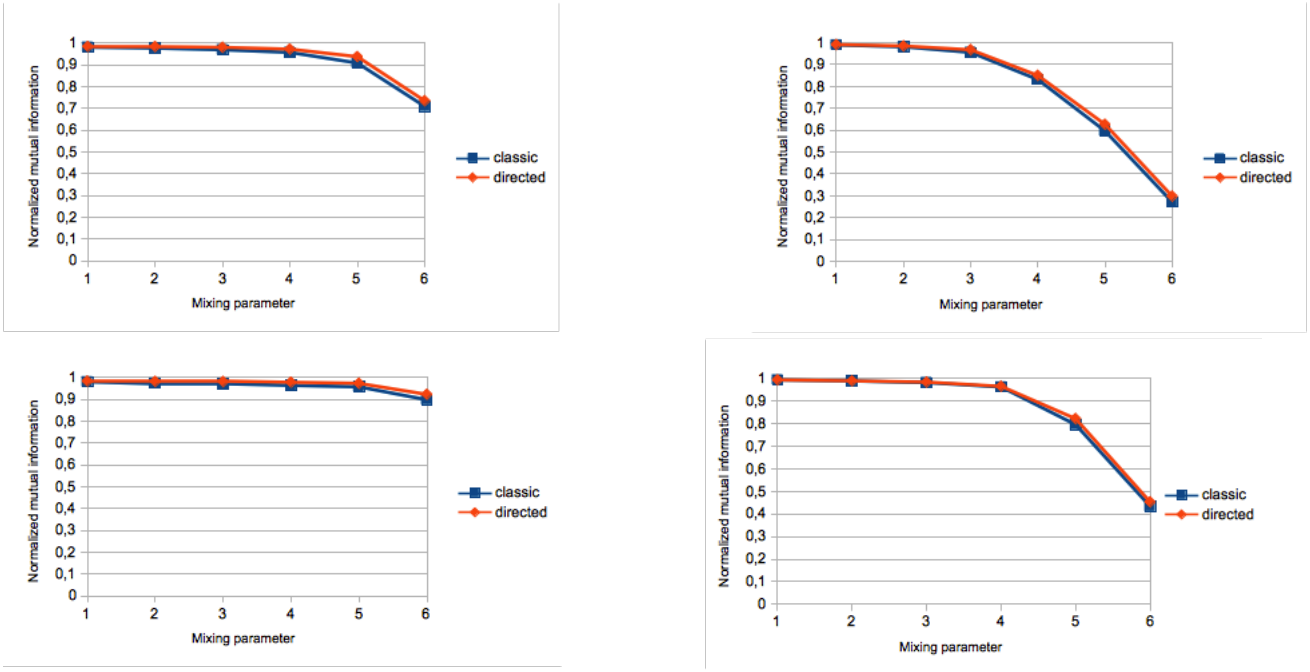

Figure 4: The values of the normalized mutual information are represented on the Y axis, while the mixing parameter is on the X axis.

degrees w.r.t. the power law and mixing parameter. For every combination of such parameters, we generated 100 different graphs and ran a statistical analysis. We first give a general picture of our observations.

n	# graphs	$\geqslant$	$\geqslant^{+0.05}$	$\geqslant^{+0.1}$	$\geqslant^{+0.25}$
100	9900	6648	2331	966	209
500	9900	6833	1193	356	33
1000	9900	6789	926	263	22
5000	9900	6674	770	211	31

Table 4: Proportion of graphs where nmi_o is greater than nmi_d by a given percentage.

It arises from Table 4 that in 70% of the cases, directed modularity provides better results than the classic Louvain's algorithm. We would like to mention that very similar results can be observed when considering other similarity measures such as F-measure, V-measure, purity, completeness, homogeneity. To be consistent with the results presented in [10], we have a closer look on the measures for 1000 and 5000 nodes, respectively. The results presented Figure 4 were obtained by taking the average of the measures over 100 different graphs with the same parameters.

We would like to mention that Louvain's algorithm seems to be particularly well-suited to maximize directed modularity, since the results obtained seem to be really better than the ones presented

by Fortunato et al. LF09a, obtained using simulated annealing for modularity optimization [6].

5.5 A comparative analysis with OSLOM [12]

To conclude this part of our work, we would like to compare Louvain's algorithm optimizing directed modularity with another algorithm used to detect communities in directed networks, namely OSLOM [12]. We are particularly interested in the *time complexity* needed to run such algorithms. We want to emphasize that such an algorithm is really not well-suited for handling large (directed) networks.

Results. We now present the results obtained by running OSLOM [12] on the classic LFR benchmarks. We focus on both the accurency and the time complexity needed to obtain such results.

n	mu	NMI	V-measure	Homogeneity	Completeness	Purity
1000	0.1	0.999	0.999	0.999	0.999	0.999
1000	0.6	0.999	0.999	0.999	0.999	0.999
5000	0.1	0.994	0.999	0.999	0.999	0.999
5000	0.6	0.999	0.999	0.999	0.999	0.999

Table 5: Results obtained on the classic LFR benchmarks. Each mesure indicates the average taken over 100 graphs with the indicated parameters.

As one can see on Table 5, OSLOM [12] is drastically better than Louvain's algorithm on the classic LFR benchmarks. This is not a surprising result, since OSLOM [12] seems to provide really good results [10]. However, as we shall see in the remaining of the paper, OSLOM [12] cannot provide results in a reasonnable amount of time for networks with relatively small size.

Time complexity. We now focus on the time complexity needed to obtain results when using OSLOM or Louvain's algorithm with directed modularity. We conduct our analysis on a set of real networks extracted from Konect Database [7] (see Table 6).

Name	Vertices	Arcs	Louvain	OSLOM
E-COLI	99	212	~ 0 seconds	1.8 seconds
BIO-YEAST	688	1079	~ 0 seconds	~ 35 seconds
SPANISH-BOOK	12643	57772	~ 0.2 seconds	~ 27 minutes
WORD-ASSOCIATION	10617	72168	~ 0.26 seconds	~ 30 minutes
EDINBURG	23219	325624	~ 1 seconds	> 10 hours

Table 6: Directed networks used for time complexity analysis.

As mentioned previously, we do not consider here the quality of the output (OSLOM performs better than Louvain's algorithm), but we only focus on the time needed to obtain such an output. For networks with 10000 vertices, OSLOM [12] already takes a significant amount of time[3] to compute the results. On the largest graph we consider, the classic configuration of OSLOM [12] fails at producing any output even after hours of computation.

Thus, if one should clearly prefer using OSLOM [12] on *small networks*, its use is absolutely impossible for networks as large as the ones that really often arise in the literature. We hope that our study will enlight that the version of Louvain's algorithm that maximizes directed modularity should be considered as more reliable to deal with such networks.

References

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: a nucleus for a web of open data. In *Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, ISWC'07/ASWC'07, pages 722–735, 2007.

[2] Vincent Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Louvain method: Finding communities in large networks. https://sites.google.com/site/findcommunities/.

[3] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *J. of Stat. Mech.: Theory and Experiment*, 2008(10):P10008, 2008.

[4] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and Krishna P. Gummadi. Measuring User Influence in Twitter: The Million Follower Fallacy. In *ICWSM '10: Proc. of int. AAAI Conference on Weblogs and Social*, 2010.

[5] Vicenç Gómez, Andreas Kaltenbrunner, and Vicente López. Statistical analysis of the social network and discussion threads in slashdot. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 645–654, 2008.

[3] We would like to mention that our results differ significantly from the ones presented in [12]. This comes from the fact that a new version of OSLOM [12] has been released. While such a version provides better results than the previous ones, it seems that the time needed to obtain the results is more important.

[6] Roger Guimera and Luis A Nunes Amaral. Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, 2005.

[7] Konect. KONECT datasets, June 2014.

[8] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E*, 80(1), 2009.

[9] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E*, 80(1):016118, 2009.

[10] Andrea Lancichinetti and Santo Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80(5), 2009.

[11] Andrea Lancichinetti and Santo Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84(6):066122, 2011.

[12] Andrea Lancichinetti, Filippo Radicchi, José J. Ramasco, and Santo Fortunato. Finding Statistically Significant Communities in Networks. *PLoS ONE*, 6(5), 2011.

[13] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Phys. Rev. Lett.*, 100, 2008.

[14] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), 2007.

[15] Alan Mislove, Hema Swetha Koppula, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Growth of the flickr social network. In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 25–30, 2008.

[16] M. E. J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256, 2003.

[17] Tore Opsahl, Filip Agneessens, and John Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3):245 – 251, 2010.

[18] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning(EMNLP-CoNLL)*, pages 410–420, 2007.

[19] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, 2009.

[20] Martin Rosvall and Carl T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[21] Alexander Strehl and Joydeep Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, 2002.

[22] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.

[23] Beichuan Zhang, Raymond Liu, Daniel Massey, and Lixia Zhang. Collecting the internet as-level topology. *SIGCOMM Comput. Commun. Rev.*, 35(1):53–61, 2005.

[24] Ying Zhao and George Karypis. Criterion functions for document clustering: Experiments and analysis. Technical report, 2002.