Parallel modularity computation for directed weighted graphs with overlapping communities[1]

[1]Mikhail Drobyshevskiy <drobyshevsky@ispras.ru>
[1]Anton Korshunov <korshunov@ispras.ru>
[1,2,3]Denis Turdakov <turdakov@ispras.ru>
[1]Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[2]Lomonosov Moscow State University,
GSP-1, Leninskie Gory, Moscow, 119991, Russia
[3]National Research University Higher School of Economics (HSE)
20 Myasnitskaya Ulitsa, Moscow, 101000, Russia

Abstract. The paper presents new versions of modularity measure for directed weighted graphs with overlapping communities. We consider several approaches to computing modularity and try to extend them. Taking into account computational complexity, we suggest two parallelized extensions which are scalable to large graphs (more than 10^4 nodes).

Keywords: modularity; community detection; PageRank; LinkRank; belonging function; belonging coefficient.

DOI: 10.15514/ISPRAS-2016-28(6)-11

For citation: Drobyshevskiy M., Korshunov A., Turdakov D. Parallel modularity computation for directed weighted graphs with overlapping communities. Trudy ISP RAN/Proc. ISP RAS, vol. 28, issue 6, 2016, pp. 153-170. DOI: 10.15514/ISPRAS-2016-28(6)-11

1. Introduction

The motivation of our research into modularity computation was the need to quantitatively assess and compare the quality of various clustering algorithms applied to mobile call graphs. As soon as no such graphs with ground-truth community structure were found, we couldn't use the most popular quality metric based on Normalized Mutual Information (NMI).

For evaluating quality of community detection methods on graphs with unknown reference communities, metrics based on probabilistic models are used. Such metrics include modularity, surprise, significance [19], ER-modularity [5]. Also, generative models from model-based community detection methods can be used to estimate likelihood of clustered graph [15, 11].

Modularity value characterizes the strength of a particular clustering of a graph. It is high when clusters are dense and sparsely connected to each other, whereas its value is low when clusters are formed at random. Besides evaluation of community cover, modularity is also used as optimization function in some community detection algorithms [16, 18]. In [12] modularity is also used for graph partitioning, but only for the case of two communities.

Here we consider modularity metric, its existing extensions for directed and weighted graphs and for the case of overlapping communities. Then we describe our extensions of modularity for overlapping communities in directed weighted graphs.

[1]This research was collaborated with and supported by Russian Research Center, Huawei Technologies.

2. Notation

In this paper we will use the following notation, most of which are common in graph theory.

$G(V, E)$ – graph with nodes V and edges E, nodes $i, j, k \in V$, edge $l(i, j) \in E$;

A – adjacency matrix of graph G;

$A_{i,j}$ – an element of A;

$w_{i,j}$ – weight of edge $l(i, j)$;

k_i – degree of node i;

C – set of communities on graph G, $c \in C$ – particular community;

C_i – set of communities node i belongs to;

S – average community size in graph G, $S = \frac{1}{|C|} \sum_{c \in C} |c|$;

Σ – average square community size in graph G, $\Sigma = \frac{1}{|C|} \sum_{c \in C} |c|^2$;

We will also use V, E, C instead of $|V|, |E|, |C|$ to denote sizes of corresponding sets.

3. Existing versions of modularity

Modularity was defined by Newman and Girvan [3] to measure a quality of a partition of a graph into a set of clusters. It is the fraction of edges within the clusters minus the expected such fraction in a randomly connected graph with the same nodes and their degrees. Modularity was originally defined for undirected unweighted graphs and is given by:

$$Q = \sum_{c \in C} \left[\frac{E_c^{in}}{E} - \left(\frac{2E_c^{in} + E_c^{out}}{2E} \right)^2 \right], \quad (1)$$

where E_c^{in} – number of edges between nodes within community c, E_c^{out} – number of edges from the nodes in community c to the nodes outside c.

Modularity can equivalently be expressed via adjacency matrix A_{ij} and nodes degrees k_i:

$$Q = \frac{1}{2E} \sum_{c \in C} \sum_{i,j \in c} \left(A_{ij} - \frac{k_i k_j}{2E} \right) \quad (2)$$

There are three main directions of extension of the original modularity definition: for directed graphs, for weighted graphs, and for the case of overlapping communities.

3.1 Modularity for directed and weighted graphs

Extension of modularity (2) to directed graphs is rather straightforward [7]:

$$Q = \frac{1}{E} \sum_{c \in C} \sum_{i,j \in c} \left(A_{ij} - \frac{k_i^{out} k_j^{in}}{E} \right), \quad (3)$$

where k_i^{out} is out-degree of node i and k_j^{in} is in-degree of node j.

Modularity (2) is easily generalized to weighted graphs as well [2]:

$$Q = \frac{1}{2m} \sum_{c \in C} \sum_{i,j \in c} \left(w_{ij} - \frac{w_i w_j}{2m} \right), \quad (4)$$

where w_{ij} – weight of edge $l(i,j)$, $w_i = \sum_j w_{ij}$ is sum of all weights of edges of node i, and $m = \frac{1}{2} \sum_{i,j} w_{ij}$ is total weight of all edges.

Moreover, modularity formula (2) for both weighted and directed graphs can be written as [6]:

$$Q = \frac{1}{m} \sum_{c \in C} \sum_{i,j \in c} \left(w_{ij} - \frac{w_i^{out} w_j^{in}}{m} \right) \quad (5)$$

Finally, modularity based on LinkRank, was suggested for weighted directed graphs [9]:

$$\begin{aligned} Q &= \sum_{c \in C} \sum_{i,j \in c} (L_{ij} - \pi_i \pi_j) \\ L_{ij} &= \pi_i G_{ij} - \text{LinkRank}, \\ \vec{\pi} &= (\pi_1, \ldots, \pi_V) - \text{PageRank vector} \end{aligned} \quad (6)$$

LinkRank is an analogy of PageRank [14] for links. PageRank is the probability of a particular page (node) being visited by a random surfer and can be defined as a stationary row vector of Google Matrix G: $\vec{\pi}^T = \vec{\pi}^T G$. In case of directed graphs Google Matrix $G_{ij} = \alpha \frac{w_{ij}}{w_i^{out}} + \frac{1}{N}(\alpha g_i + 1 - \alpha)$, where α is damping parameter for PageRank (with probability $1 - \alpha$ random surfer jumps to a random node) and g_i is indicator of dangling node:

$$g_i = \begin{cases} 1 & \text{if node is dangling } (w_i^{out} = 0) \\ 0 & \text{otherwise} \end{cases}$$

This formula originates from an alternative notion of community as a group of nodes where a random surfer spends more time in average. More technically, this definition of modularity is the deviation between the fraction of time a random walker spends within communities and the expected such time.

3.2 Overlapping modularity

In the case when a node can belong to several communities, the belonging coefficients $a_{i,c}$ are introduced [8] which indicate how much a node i belongs to community c. This coefficients are non-negative and sum to one: $\forall i \in V, \forall c \in C \quad a_{i,c} > 0 \quad \sum_{c \in C} a_{i,c} = 1$. This relates to another extension of community detection problem, called fuzzy community detection [13]. To generalize different approaches of using belonging coefficients, a belonging function $f(a_{i,c_i}, a_{j,c_j})$ can be defined [17] to characterize an extent to what an edge (i,j) connects communities c_i and c_j respectively.

According to this, several approaches for overlapping modularity from the literature can be generalized to the following two definitions [17]:

$$\begin{aligned} Q &= \sum_{c \in C} \left[\frac{E_c^{in}}{E} - \left(\frac{2E_c^{in} + E_c^{out}}{2E} \right)^2 \right] \\ E &= \frac{1}{2} \sum_{i,j \in V} A_{ij} \\ E_c^{in} &= \frac{1}{2} \sum_{i,j \in c} A_{ij} \cdot f(a_{i,c}, a_{j,c}) \\ E_c^{out} &= \sum_{i \in c} \sum_{j \in c' \neq c} A_{ij} \cdot f(a_{i,c}, a_{j,c'}) \end{aligned} \quad (7)$$

and

$$Q = \frac{1}{2E} \sum_{c \in C} \sum_{i,j \in c} \left(A_{ij} - \frac{k_i k_j}{2E} \right) f(a_{i,c}, a_{j,c}) \quad (8)$$

where belonging coefficient can be:

$$a_{i,c} = \begin{bmatrix} \frac{1}{C_i} \\ \frac{\sum_{k \in c} A_{ik}}{\sum_{c' \in C_i} \sum_{k \in c'} A_{ik}} \\ \frac{\sum_{k \in c} \frac{M_{ik}^c}{M_{ik}} A_{ik}}{\sum_{c \in C} \sum_{k \in c} \frac{M_{ik}^c}{M_{ik}} A_{ik}} \end{bmatrix}, \quad (9)$$

where M_{ik} is the number of maximal cliques containing edge (i, j), M_{ik}^c is the number of maximal cliques containing edge (i, j) inside community c. Belonging function can be:

$$f(a, b) = \begin{bmatrix} \frac{a+b}{2} \\ ab \\ max(a, b) \end{bmatrix} \quad (10)$$

3.3 Further extensions of modularity

Besides the node-based extensions, there was suggested edge-based extension [10] (for directed graphs):

$$Q = \frac{1}{E} \sum_{c \in C} \sum_{i,j \in V} \left(\beta_{l(i,j),c} A_{ij} - \frac{\beta^{out}_{l(i,\cdot),c} k_i^{out} \beta^{in}_{l(\cdot,j),c} k_j^{in}}{E} \right)$$

$$\beta_{l(i,j),c} = f(a_{i,c}, a_{j,c}) - \text{edge belonging coefficient},$$

$$\beta^{out}_{l(i,\cdot),c} = \frac{1}{V} \sum_{k \in V} f(a_{i,c}, a_{k,c}) - \text{expected of that for outcoming link}, \quad (11)$$

$$\beta^{in}_{l(\cdot,j),c} = \frac{1}{V} \sum_{k \in V} f(a_{k,c}, a_{j,c}) - \text{expected of that for incoming link}.$$

Here edge belonging function $f(a_{i,c}, a_{j,c})$ can be any of (10), but the authors suggested this variant (together with empirically found expression for $h(x)$):

$$f(a, b) = \frac{1}{(1 + e^{-h(a)})(1 + e^{-h(b)})}, \quad h(x) = 2px - p, \;\; p = 30. \quad (12)$$

It is worth to notice that actually in the inner sum iterating of pairs of nodes i, j are done over nodes only from community c (not from the whole V), due to the form of β functions.

Authors of [17] suggested density-based version of modularity (1) for overlapping directed graphs:

$$Q_D = \sum_{c \in C} \left[\frac{E_c^{in}}{E} \cdot d_c - \left(\frac{2E_c^{in} + E_c^{out}}{2E} \cdot d_c \right)^2 - \sum_{c' \neq c} \frac{E_{c,c'}}{2E} d_{c,c'} \right]$$

$$d_c = \frac{2E_c^{in}}{\sum_{i \neq j \in c} f(a_{i,c}, a_{j,c})} - \text{internal density}$$

$$d_{c,c'} = \frac{E_{c,c'}}{\sum_{i \in c, j \in c'} f(a_{i,c}, a_{j,c'})} - \text{pair-wise density} \quad (13)$$

$$E_{c,c'} = \sum_{i \in c, j \in c'} A_{ij} \cdot f(a_{i,c}, a_{j,c'})$$

3.4 Drawbacks and limitations

The first obvious drawback is that there was not found any modularity formula, comprising all three needed properties: support of directed, weighted graphs with overlapping communities.

The second limitation is computational complexity. Aforementioned formulas of overlapping modularity are not acceptable for large graphs (with more than 10^4 nodes within community cover) due to their high computational complexity. Denoting the average number of communities by C, average community size by S and number of nodes by V, we have for (13) time complexity $O(C^2 \Sigma)$, and for (11) — $O(CV^2 \Sigma)$. See subsection 4.1 for more details.

It's also worth noting that LinkRank authors [9] provide some evidence that the modularity (5) can't distinguish the direction of links.

4. Our extensions of modularity

Since we focus on modularity for directed weighted graphs with overlapping communities, we actually have two possibilities of extension: make overlapping (directed) modularities support weights, or to extend directed weighted modularities to the overlapping case.

The first approach suggests naive substitution of adjacency matrix of a graph to matrix of weights and number of edges to the sum of their weights. Doing so with density formula (13) leads to unnormalization: values of modularity start to exceed the available range $(-\frac{1}{2}; 1]$. But we will still use it in experiments with unweighted graphs. On the other hand, edge-based formula seems to allow such generalization, becoming:

$$Q_E = \frac{1}{m} \sum_{c \in C} \sum_{i,j \in V} \left(\beta_{l(i,j),c} w_{ij} - \frac{\beta^{out}_{l(i,\cdot),c} w_i^{out} \beta^{in}_{l(\cdot,j),c} w_j^{in}}{m} \right) \quad (14)$$

But this is still computationally expensive.

The second approach consists in introducing belonging coefficients (9) and belonging functions (10) to simple version (5):

$$Q_S = \frac{1}{m} \sum_{c \in C} \sum_{i,j \in c} \left(w_{ij} - \frac{w_i^{out} w_j^{in}}{m} \right) f(a_{i,c}, a_{j,c}), \quad (15)$$

and to LinkRank-based version of modularity (6):

$$Q_{LR} = \sum_{c \in C} \sum_{i,j \in c} (L_{ij} - \pi_i \pi_j) f(a_{i,c}, a_{j,c}). \quad (16)$$

Since PageRank (and hence LinkRank) has fast implementations ([1, 4]), these two formulas have much lower computational complexities.

Also, we suggested to use in formulas a normalization coefficient instead of belonging function:

$$f(a_{i,c}, a_{j,c}) \mapsto \frac{1}{|C_i \cap C_j|} \quad (17)$$

The intuition is the following. If both nodes i, j belong to n communities, the term $\sum_{i,j \in c} (...)$ will encounter n times in modularity formula, once for each community, so we weigh it by the factor of $\frac{1}{n} = \frac{1}{|C_i \cap C_j|}$. It's easy to see that otherwise modularity can become unlimited: suppose that each community is actually two equal different communities, then modularity value doubles.

4.1 Computational complexity

Here we calculate computational complexities of modularity extensions Q_D, Q_E, Q_S and Q_{LR}. All complexities are present in table 1.

Firstly, denote by $O(F)$ computational complexity of $f(a_{i,c}, a_{j,c})$ – we consider it later.

In the expression for Q_D (13), the term E_c^{out} is computed in $O(C\Sigma F)$, so as d_c; E_c^{out} in $O(C\Sigma F)$; $E_{c,c'}$ and $d_{c,c'}$ in $O(S^2 F)$ time. Counting that average square community size Σ is not less than square of average size S^2, each term of summation has complexity $O(C\Sigma F)$, giving overall complexity $O(C^2 \Sigma F)$.

In the expression for Q_E (14), the hardest term is β^{in} and β^{out}, which take $O(VF)$ steps, thus resulting in $O(CV^2F^2)$ overall complexity.

Q_S (15) and Q_{LR} (16) have complexity $O(C\Sigma F)$, ignoring PageRank calculation time as insignificant. Understanding the big-O complexity of PageRank calculation requires analyzing the code of pagerank scipy method from NetworkX[2]. However, Aric Hagberg (NetworkX Lead Programmer) wrote that their implementation has

[2] http://networkx.github.io/

"linear complexity in the number of edges". In practice, PageRank computation time is negligible.

Now consider $f(a_{i,c}, a_{j,c})$. Uniform belonging coefficient $a_{i,c} = \frac{1}{C_i}$ may be computed by one operation if communities for each node are explicitly known, e.g. each node has a set of labels. But usually community detection algorithms return list of communities represented by sets of nodes. This means we need $O(C \log S)$ operations to find all communities a given node i belongs to. The same concerns fraction belonging coefficient, for which we have $O(S + C \log S)$, supposing that average node membership is not very high, i.e. $C_i = O(1)$. Therefore, intersection belonging function together with the others are $O(1)$.

Table 1: Computational complexities for modularity formulas, belonging functions and belonging coefficients.

formula	**complexity**
Q_D	$O(C^2\Sigma)$
Q_E	$O(CV^2\Sigma)$
Q_S	$O(C\Sigma)$
Q_{LR}	$O(C\Sigma)$
belonging function	
sum	$O(1)$
product	$O(1)$
intersection	$O(1)$
edge-based	$O(1)$
belonging coefficient	
uniform	$O(C \log S)$
fraction	$O(S + C \log S)$

4.2. Effects

In order to demonstrate adequacy of the estimate based on computed modularity with regard to intuitive community structure, we computed modularities of several alternative community covers of the example graph (see Fig. 1). We generated a large set of random community covers, and sort them according to the modularity value computed with formula (15). Fig. 1 demonstrates 3 covers with highest modularity and 3 covers with lowest modularity. We can see that the most intuitive cover corresponds to the highest modularity value. The same holds for formula (16).

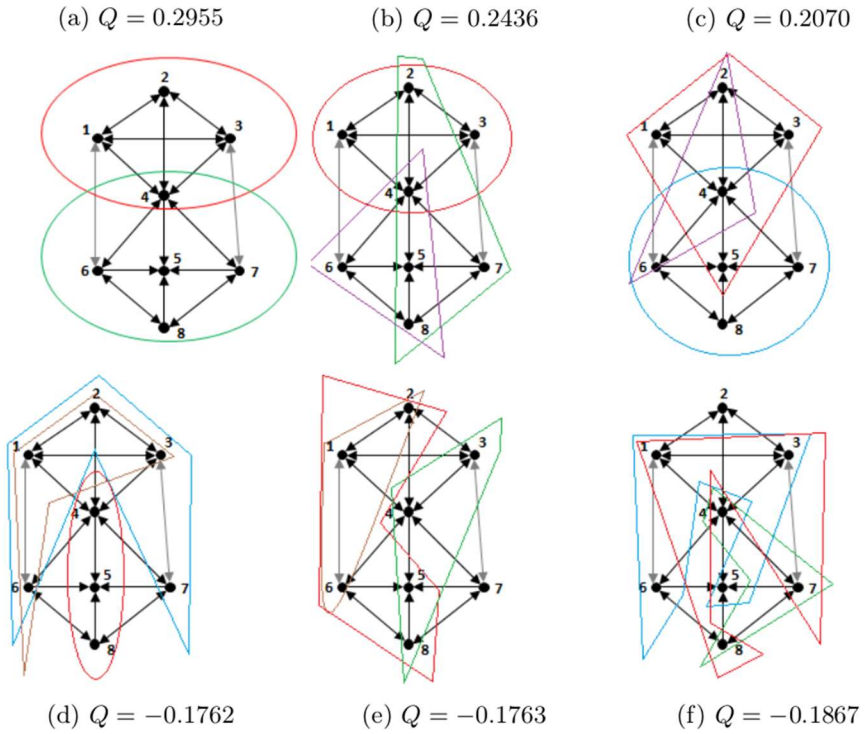

Fig. 1. Modularity (15) of different community covers for example directed weighted graph. All edges have weight 10 except (1-6) and (3-7) which have weight 0.1. Top row: covers with maximal modularity; bottom row: covers with minimal modularity.

5. Experiments

We implemented in Python four versions of modularity Q_D, Q_E, Q_S and Q_{LR} together with 4 belonging functions (see (10) and (12)):

1. sum $f(a,b) = \frac{1}{2}(a+b)$,
2. product $f(a,b) = ab$,
3. intersection $f(a,b) = \dfrac{1}{|C_i \cup C_j|}$,
4. edge-based $f(a,b) = (1+e^{-60a+30})^{-1}(1+e^{-60b+30})^{-1}$,

and two belonging coefficients:

1. uniform $a_{i,c} = \dfrac{1}{C_i}$,
2. fraction $a_{i,c} = \dfrac{\sum_{k \in c} A_{ik}}{\sum_{c' \in C_i} \sum_{k \in c'} A_{ik}}$.

Also we conducted a set of experiments: on computation time, different belonging functions and belonging coefficients, and parallelizing.

5.1 Computation time

We compared modularity value and computation time of four appropriate formulas (Q_D, Q_E, Q_S, Q_{LR}) on two graphs of different size. Since Q_D doesn't support weights and fraction belonging coefficient is undefined for directed graphs (due to possible zero in denominator), graphs were chosen undirected unweighted. Experiments with directed weighted graphs are to be conducted later. We took default belonging functions (suggested in original papers) and uniform belonging coefficient for simplicity.

The small graph was generated by CDR-GEN generator[3] and clustered by SLPA algorithm[4] with threshold $p = 0.10$. Parameters of the smaller graph are: number of nodes $|V| = 3124$, number of edges $|E| = 3913$, number of communities $|C| = 333$, average size of community $S = 10.1$ with 100% of nodes involved in communities, average membership 1.14.

The big graph was Wu et al dataset[5] clustered by MOSES algorithm[6] with 7% of nodes involved in communities ($|V| = 72111, |E| = 79003, |C| = 899$, $S = 5.2$). Results are in table 2.

Table 2 shows that as size of graph and size and number of communities grow, Q_D and Q_E become too computationally expensive, so there are only two scalable candidates, Q_S and Q_{LR}.

Table 2: Modularity value and computation time for Q_S, Q_{LR}, Q_D and Q_E on 2 undirected unweighted graphs.

formula	bel.func.	bel.coef.	complexity	Time small	Time big
Q_D	product	uniform	$O(C^3 \Sigma \log S)$	3m57s	3h12m
Q_E	edge-based	uniform	$O(C^2 V^2 \Sigma \log S)$	7m17s	53h53m
Q_S	intersection	uniform	$O(C^2 \Sigma \log S)$	0.6s	18s
Q_{LR}	intersection	uniform	$O(C^2 \Sigma \log S)$	0.7s	18s

[3] https://github.com/mayconbordin/cdr-gen
[4] https://sites.google.com/site/communitydetectionslpa/
[5] http://www.pnas.org/content/107/44/18803?tab=ds
[6] https://sites.google.com/site/aaronmcdaid/moses

Table 3: Comparison of different belonging functions and belonging coefficients for Q_D, Q_E, Q_S and Q_{LR} on undirected unweighted graph with $|V| = 72146$, $|E| = 79003$, $|C| = 1894$, $S = 5.30$ (clustered by Clique Percolation).

formula	bel.coef.	sum	product	intersection	edge-based
Q_D	uniform	0.292 (2481 s)	0.291 (2425 s)	undefined	0.291 (2266 s)
Q_D	fraction	0.292 (2355 s)	0.292 (2187 s)	undefined	0.293 (2752 s)
Q_E	uniform	0.737 (4688 s)	0.714 (3201 s)	0.763 (1488 s)	0.713 (3349 s)
Q_E	fraction	0.740 (4130 s)	0.718 (4795 s)	0.763 (1476 s)	0.719 (3683 s)
Q_S	uniform	0.737 (4 s)	0.714 (3 s)	0.760 (1 s)	0.713 (2 s)
Q_S	fraction	0.738 (3 s)	0.715 (3 s)	0.760 (1 s)	0.717 (3 s)
Q_{LR}	uniform	0.647 (4 s)	0.628 (4 s)	0.665 (2 s)	0.628 (4 s)
Q_{LR}	fraction	0.647 (5 s)	0.630 (5 s)	0.665 (2 s)	0.631 (8 s)

Table 4: Comparison of different belonging functions for Q_S and Q_{LR} on a directed weighted graph with overlapping communities ($|V| = 72146$, $|E| = 79003$, $|C| = 1894$, $S = 5.30$, clustered by Clique Percolation).

	$\|\mathbf{C}\| = \mathbf{1894}$, $\mathbf{S} = \mathbf{5.30}$ (Clique Percolation)			$\|\mathbf{C}\| = \mathbf{6731}$, $\mathbf{S} = \mathbf{9.05}$ (SLPA)		
formula	**sequential**	**parallel 1**	**parallel 2**	**sequential**	**parallel 1**	**parallel 2**
Q_S	104s	37s	38s	106m	36m	32m
Q_{LR}	105s	39s	40s	103m	37m	32m

Table 5: Comparison of times of sequential and parallel versions (N = 6 processes) of Q_S and Q_{LR} on a directed unweighted graph with $|V| = 72146$, $|E| = 79003$ clustered by Clique Percolation (covers 13% of nodes) and SLPA (covers 78% of nodes) algorithms.

	$\|\mathbf{C}\| = \mathbf{1894}$, $\mathbf{S} = \mathbf{5.30}$ (Clique Percolation)			$\|\mathbf{C}\| = \mathbf{6731}$, $\mathbf{S} = \mathbf{9.05}$ (SLPA)		
formula	**sequential**	**parallel 1**	**parallel 2**	**sequential**	**parallel 1**	**parallel 2**
Q_S	104s	37s	38s	106m	36m	32m
Q_{LR}	105s	39s	40s	103m	37m	32m

5.2 Belonging functions and belonging coefficients

Then we investigated influence of different belonging functions and belonging coefficients on values of Q_S and Q_{LR}. We used the same Wu et al dataset clustered by Clique Percolation algorithm[7] with 13% of nodes involved in communities ($|V| = 72146, |E| = 79003, |C| = 1894, S = 5.30$).

Table 3 shows that the choice of belonging function or belonging coefficient doesn't make much difference to result modularity. Meanwhile, intersection belonging function takes the lowest time. Values of Q_S are in good consistency with those of Q_E, which is widely used in papers. Q_{LR} values tend to be less than Q_S and Q_E. Q_D values differ a lot, possibly due to dissimilar formula structure, but as far as we know this formula was not compared to other ones in literature.

Table 4 extends the comparison of different belonging functions for Q_S and Q_{LR} on a directed weighted graph with overlapping communities. Belonging coefficient is

[7] http://www.cfinder.org/

uniform. We see that the behavior is consistent with that of undirected unweighted case.

5.3 Parallel modularity

Computation process of Q_S and Q_{LR} naturally allows parallelization. Since each community and each node pair contributes independently to the modularity value, iterating over node pairs may be distributed between processors.

We implemented two parallel versions. The first one is rather straightforward. Iteration over communities is left sequential. Each time when community of size more than $c_0 = 100$ is encountered, $N = 6$ parallel processes are initialized. The set of all nodes pairs within the community is split into N equal chunks and are assigned to these processes (see algorithm 1).

```
for c ∈ C do
    if |c| > c_0 then
        {c_1, ..., c_N} ← split c into N equal
          chunks
        do in parallel i ∈ 1, N:
          computeModularity(c_i)
    end
end
```

Algorithm 1: *Parallel modularity version 1.*

The second parallel version is a little more complicated. The idea is to split the set of communities C into subsets between processors. But in order to balance the load, these chunks should have approximately equal sum of squares of community size since community of size s has s^2 ordered node pairs (counting self-loops). To achieve this we used a greedy algorithm, which iterates over communities in descending order and assigns each of them to a subset that has the smallest sum of size squares. The only problem here is that the biggest community may have size square much more than sum of size squares of the rest ones, i.e. the chunk which gets this community will be overloaded. To overcome this challenge we sort communities by their sizes in descending order and apply the first parallel approach to first (biggest) several communities, until we encounter community with small enough size to allow balancing of the rest ones or reach lower community size bound c_0. The rest ones are split into subsets according to the mentioned greedy algorithm. To determine whether to start balancing we use a simple condition: square of size of current biggest community should be at most $\frac{1}{N}$ of total sum of squares of sizes of communities left at the moment. Formally, having sorted sizes of communities $s_1 \geqslant s_2 \geqslant ... \geqslant s_C$, the condition of stopping at community k is $s_k^2 \leqslant \frac{1}{N}(s_k^2 + ... + s_C^2)$. See algorithm 2.

sortBySizeInDescendingOrder(C)

for $c_i \in C$ **do**
  if $|c_k| \leqslant c_0$ *or* $|c_k|^2 \leqslant \frac{1}{N}(|c_k|^2 + ... + |c_{|C|}|^2)$ **then**
    break
  end
  $C \leftarrow C \setminus \{c_k\}$
  $\{c_1, \ldots, c_N\} \leftarrow$ split c into N equal chunks
  do in parallel $i \in \overline{1, N}$:
    computeModularity(c_i)
end
$\{C_1, ..., C_N\} \leftarrow$ balanceSumOfSquares(C)
do in parallel $i \in \overline{1, N}$: **for** $c_{ik} \in C_i$ **do**
  computeModularity(c_{ik})
end

Algorithm 2: *Parallel modularity version 2.*

We compared the speedup due to both versions of parallellization versus sequential computing of modularity for Q_S and Q_{LR}. See table 5. When number of communities is small ($|C| = 1894$) the first method is slightly faster due to its simplicity (results were averaged over 5 runs). In case of many communities the second version shows its benefit.

We also investigated process scalability of both parallel implementations. The results are represented in Fig. 1.

6. Conclusion

We investigated existing approaches to computing modularity measure and developed Q_S and Q_{LR} – modularity extensions for large directed weighted graphs with overlapping communities. These extensions have low computational complexity which makes them applicable to graphs with more than 10^4 nodes and they also can be computed in parallel way.

These two formulae are based on different notions of community: as group of nodes with more dense links (Q_S) or a group of nodes where a random surfer tends to spend more time (Q_{LR}). Since a surfer walks along link direction, the second formula is more sensible to direction of links in a graph.

As a future direction may be considered a possibility to use new version of modularity for overlapping community detection in directed weighted graphs.

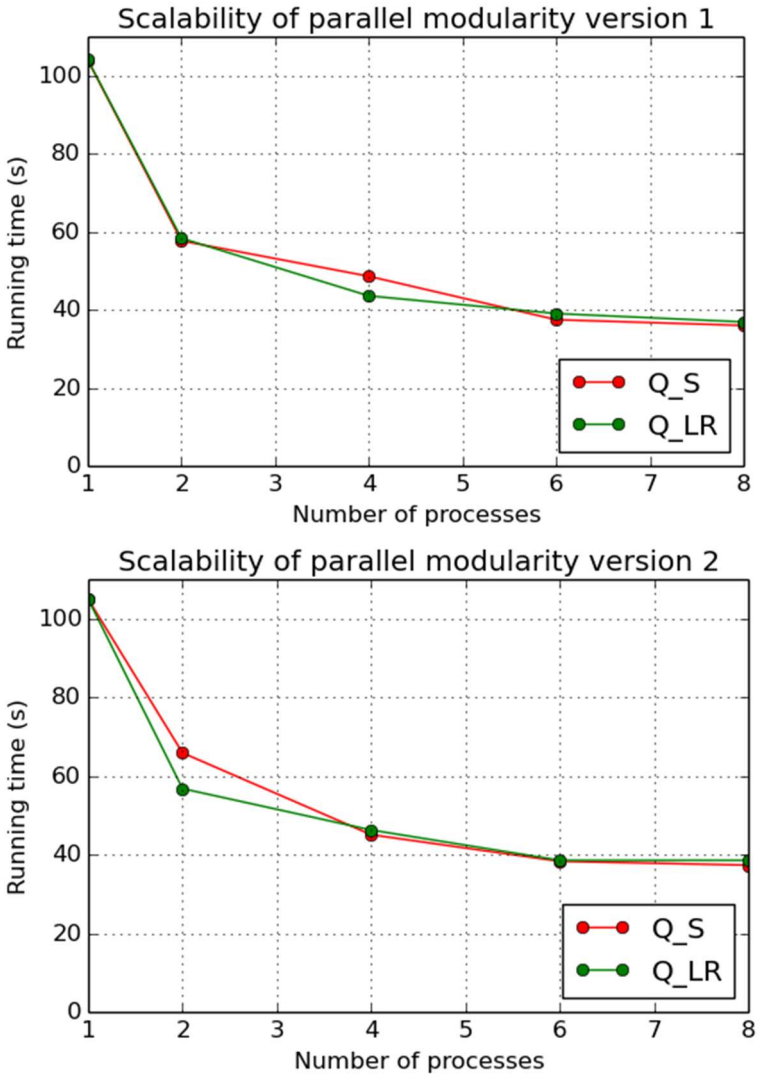

Fig. 2. Speedup of both parallel modularity versions on a directed unweighted graph with |V| = 72146, |E| = 79003, |C| = 1894, S = 5.30

References

[1]. Lawrence Page et al. “The PageRank citation ranking: bringing order to the web.” Technical Report. Stanford InfoLab. (1999).

[2]. Mark EJ Newman. “Analysis of weighted networks”. *Physical Review E* 70.5 (2004), p. 056131.

[3]. Mark EJ Newman, Michelle Girvan. “Finding and evaluating community structure in networks”. *Physical review E* 69.2 (2004), p. 026113.
[4]. Amy N Langville, Carl D Meyer. “A survey of eigenvector methods for web information retrieval”. *SIAM review* 47.1 (2005), pp. 135–161.
[5]. Jörg Reichardt, Stefan Bornholdt. “Statistical mechanics of community detection”. *Physical Review E* 74.1 (2006), p. 016110.
[6]. Alex Arenas et al. “Size reduction of complex networks preserving modularity”. *New Journal of Physics* 9.6 (2007), p. 176.
[7]. Elizabeth A Leicht, Mark EJ Newman. “Community structure in directed networks”. *Physical review letters* 100.11 (2008), p. 118703.
[8]. Tam´as Nepusz et al. “Fuzzy communities and the concept of bridgeness in complex networks”. *Physical Review E* 77.1 (2008), p. 016107.
[9]. Youngdo Kim, Seung-Woo Son, Hawoong Jeong. “Finding communities in directed networks”. *Physical Review E* 81.1 (2009), p. 016103.
[10]. Vincenzo Nicosia et al. “Extending the definition of modularity to directed graphs with overlapping communities”. *Journal of Statistical Mechanics: Theory and Experiment* 2009.03 (2009), p. 03024.
[11]. Aaron McDaid, Neil Hurley. “Detecting highly overlapping communities with model-based overlapping seed expansion”. *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*. IEEE. 2010, pp. 112–119.
[12]. Yu-Teng Chang, Dimitrios Pantazis, Richard M Leahy. “Partitioning directed graphs based on modularity and information flow”. *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*. IEEE. 2011, pp. 1105–1108.
[13]. Steve Gregory. “Fuzzy overlapping communities in networks”. *Journal of Statistical Mechanics: Theory and Experiment* 2011.02 (2011), p. 02017.
[14]. Amy N Langville, Carl D Meyer. *Google’s PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2011.
[15]. Jaewon Yang, Jure Leskovec. “Community-affiliation graph model for overlapping network community detection”. *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE. 2012, pp. 1170–1175.
[16]. Mingming Chen, Konstantin Kuzmin, Boleslaw K Szymanski. “Community detection via maximization of modularity and its variants”. *Computational Social Systems, IEEE Transactions on* 1.1 (2014), pp. 46– 65.
[17]. Mingming Chen, Konstantin Kuzmin, Boleslaw K Szymanski. “Extension of modularity density for overlapping community structure”. *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*. IEEE. 2014, pp. 856–863.
[18]. Nicolas Dugué, Anthony Perez. “Directed Louvain: maximizing modularity in directed networks”. PhD thesis. Universit´e d’Orléans, 2015.
[19]. Vincent A Traag, Rodrigo Aldecoa, J-C Delvenne. “Detecting communities using asymptotical surprise”. *Physical Review E* 92.2. APS, 2015, p. 022816.

Параллельное вычисление модулярности для направленных взвешенных графов с пересекающимися сообществами

[1]Дробышевский Михаил <drobyshevsky@ispras.ru>
[1]Коршунов Антон <korshunov@ispras.ru>
[1,2,3]Турдаков Денис <turdakov@ispras.ru>
[1]Институт системного программирования РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[2]Московский государственный университет имени М.В. Ломоносова,
119991, Россия, Москва, Ленинские горы, д. 1
[3]Национальный исследовательский университет «Высшая школа экономики»
101000, Россия, Москва, ул. Мясницкая, д. 20

Аннотация. В статье представлены новые алгоритмы расчета модулярности для направленных взвешенных графов с пересекающимися сообществами. Рассматриваются несколько подходов для вычисления модулярности и их расширения. Учитывая вычислительную сложность известных подходов, предлагаются два параллельных расширения, масштабируемых на графы с более 10^4 вершин.

Ключевые слова: модулярность; поиск сообществ; пэйдж-ранк; линк-ранк; функция принадлежности; коэффициент принадлежности.

DOI: 10.15514/ISPRAS-2016-28(6)-11

Для цитирования: Дробышевский М., Коршунов А., Турдаков Д. Параллельное вычисление модулярности для направленных взвешенных графов с пересекающимися сообществами. Труды ИСП РАН, том 28, вып. 6, 2016 г., стр. 153-170 (на английском). DOI: 10.15514/ISPRAS-2016-28(6)-11

Список литературы

[1]. Lawrence Page и др. “The PageRank citation ranking: bringing order to the web”. Technical Report. Stanford InfoLab. (1999).
[2]. Mark EJ Newman. “Analysis of weighted networks”. *Physical Review E* 70.5 (2004), стр. 056131.
[3]. Mark EJ Newman, Michelle Girvan. “Finding and evaluating community structure in networks”. *Physical review E* 69.2 (2004), стр. 026113.
[4]. Amy N Langville, Carl D Meyer. “A survey of eigenvector methods for web information retrieval”. *SIAM review* 47.1 (2005), стр. 135–161.
[5]. Jörg Reichardt, Stefan Bornholdt. “Statistical mechanics of community detection”. *Physical Review E* 74.1 (2006), стр. 016110.

[6]. Alex Arenas et al. “Size reduction of complex networks preserving modularity”. *New Journal of Physics* 9.6 (2007), стр. 176.

[7]. Elizabeth A Leicht, Mark EJ Newman. “Community structure in directed networks”. *Physical review letters* 100.11 (2008), стр. 118703.

[8]. Tamas Nepusz et al. “Fuzzy communities and the concept of bridgeness in complex networks”. *Physical Review E* 77.1 (2008), стр. 016107.

[9]. Youngdo Kim, Seung-Woo Son, Hawoong Jeong. “Finding communities in directed networks”. *Physical Review E* 81.1 (2009), стр. 016103.

[10]. Vincenzo Nicosia et al. “Extending the definition of modularity to directed graphs with overlapping communities”. *Journal of Statistical Mechanics: Theory and Experiment* 2009.03 (2009), стр. 03024.

[11]. Aaron McDaid, Neil Hurley. “Detecting highly overlapping communities with model-based overlapping seed expansion”. *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*. IEEE. 2010, стр. 112–119.

[12]. Yu-Teng Chang, Dimitrios Pantazis, Richard M Leahy. “Partitioning directed graphs based on modularity and information flow”. *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*. IEEE. 2011, стр. 1105–1108.

[13]. Steve Gregory. “Fuzzy overlapping communities in networks”. *Journal of Statistical Mechanics: Theory and Experiment* 2011.02 (2011), стр. 02017.

[14]. Amy N Langville, Carl D Meyer. “Google’s PageRank and beyond: The science of search engine rankings”. Princeton University Press, 2011.

[15]. Jaewon Yang, Jure Leskovec. “Community-affiliation graph model for overlapping network community detection”. *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE. 2012, стр. 1170–1175.

[16]. Mingming Chen, Konstantin Kuzmin, Boleslaw K Szymanski. “Community detection via maximization of modularity and its variants”. *Computational Social Systems, IEEE Transactions on* 1.1 (2014), стр. 46– 65.

[17]. Mingming Chen, Konstantin Kuzmin, Boleslaw K Szymanski. “Extension of modularity density for overlapping community structure”. *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*. IEEE. 2014, стр. 856–863.

[18]. Nicolas Dugué, Anthony Perez. “Directed Louvain: maximizing modularity in directed networks”. PhD thesis. Universit´e d’Orléans, 2015.

[19]. Vincent A Traag, Rodrigo Aldecoa, J-C Delvenne. “Detecting communities using asymptotical surprise”. *Physical Review E* 92.2. APS, 2015, стр. 022816.