Preprints, 12th IFAC International Workshop on Adaptation and Learning in Control and Signal Processing, June 29 - July 1, 2016. Eindhoven, The Netherlands

A New Randomized Algorithm for Community Detection in Large Networks ⋆

Ilia Kirianovskii * **Oleg Granichin** *,**
Anton Proskurnikov *,**,***

* *Saint Petersburg State University (Faculty of Mathematics and Mechanics, and Research Laboratory for Analysis and Modeling of Social Processes), 7-9, Universitetskaya nab., St. Petersburg, 199034, Russia*
** *Institute of Problems of Mechanical Engineering (IPME RAS), 61, Bolshoy pr., St. Petersburg, 199178, Russia and ITMO University, 14A, Birzhevaya liniya, St. Petersburg, 199034, Russia*
*** *Engineering and Technology Institute Groningen, University of Groningen, The Netherlands*

Abstract: The problem of *community detection* (or *clustering*) in graphs plays an important role in analysis of complex large-scale networks and big data structures, arising in natural, behavioral and engineering sciences. Examples of such networks include, but are not limited to, World Wide Web (WWW) and Internet, social networks, ecological networks and food webs, cellular and molecular ensembles. A community (or a module) in a graph is a subset of its nodes, whose members are "densely" connected to each other yet have relatively few connections with nodes outside this subset. A number of algorithms to subdivide the nodes of large-scale graphs into communities have recently been proposed; many of them hunt for the graph's partitions of maximal *modularity*. One of the most efficient graph clustering algorithms of this type is the Multi-Level Aggregation (or "Louvain") method. In this paper, a randomized counterpart of this algorithm is proposed, which provides a comparable "quality" of graph's clustering, being however much faster on huge graphs. We demonstrate the efficiency of our algorithm, comparing its performance on several "benchmark" large-scale graphs with existing methods.

Keywords: Networked systems, distributed parameter systems, sequential learning.

1. INTRODUCTION

Many complex systems, arising in natural, behavioral and engineering sciences, are naturally representable by *graphs* or *networks*, where the nodes stand for elementary units of the system and the arcs describe their relations or ties. The examples include, but are not limited to, World Wide Web (WWW) and Internet, collaboration and citation networks, ecological networks and food webs, cellular and molecular ensembles, continental power grids and road networks, see e.g. Albert and Barabási (2002); Newman and Girvan (2004); Newman (2006) and references therein.

Many real-world networks are known to have *community structure*, that is, their nodes "are joined together in tightly knit groups, between which there are only looser connections" (Girvan and Newman, 2002). Such groups of nodes are referred to as *communities* in the graph; communities are sometimes called *clusters*, *modules* or *compartments* (Malliaros and Vazirgiannis, 2013). Recognizing communities in a general graph is computationally difficult task, called *community detection* or *graph clustering*. Community detection problems have been extensively studied; the relevant algorithms, historical milestones and applications are reported in recent surveys (Porter et al., 2009; Fortunato, 2010; Malliaros and Vazirgiannis, 2013).

The knowledge of the community structure helps to contain the propagation of worms and viruses in computer networks and online social media (Lu et al., 2015), and identify topically related webpages in the web graphs (Kleinberg and Lawrence, 2001; Flake et al., 2002). In metabolic (Ravasz et al., 2002) and neural networks (Deco and Corbetta, 2011), communities stand for *functional units*, responsible for the same group of functions; their analysis allows to reveal the functional organizations of metabolic pathways and the nervous system. One of the first works on community detection (Rice, 1927) was concerned with voting blocs in political bodies.

The huge scale of graphs, arising in such applications as social media, makes impossible their efficient storage and mining without special preprocessing, substantially reducing the volume of information. Community structures in graphs are intimately related with data compression (Rosvall and Bergstrom, 2008; Lim et al., 2014). This is not surprising since the communities contain a lot of "redun-

⋆ The results of the paper have been obtained at IPME RAS under support of Russian Foundation for Basic Research (RFBR) grant 16-07-00890. E-mails: ki.stfu@gmail.com;o.granichin@spbu.ru;avp1982@gmail.com

Copyright © 2016 IFAC

dant" information, their nodes have similar properties and often may treated as a single object. Dense graphs of the communities are usually well compressed (Lim et al., 2014) unlike the sparse adjacency matrix of the original graph.

Most of approaches to graph clustering methods can be classified into three major groups. Algorithm of the first kind (Girvan and Newman, 2002; Newman and Girvan, 2004) segregate communities, sequentially removing the edges among them. Procedures of the second type (Pons and Latapy, 2006) "build" large communities by merging smaller "subcommunities". The third group of methods hunts for a partition of the graph's nodes, which is optimal with respect to some objective function (Blondel et al., 2008; Lancichinetti and Fortunato, 2009).

A convenient cost function, measuring the "quality" of the graph clustering, was proposed by Newman and Girvan (Newman and Girvan, 2004) and is referred to as *modularity*. The idea of modularity maximization lies in the heart of many efficient algorithms for community detection (Newman, 2004b; Clauset et al., 2004; Duch and Arenas, 2005; Schuetz and Caflisch, 2008; Blondel et al., 2008; Ovelgönne et al., 2010). Among them is a method, proposed by a group of researchers from the Catholic University of Louvain (UCL) and known as the *Louvain method* (Blondel et al., 2008). This algorithm proves to be faster on huge graphs with millions and billions of edges than many other algorithms (Clauset et al., 2004; Pons and Latapy, 2006; Wakita and Tsurumi, 2007); on typical "sparse" graphs it runs in nearly linear time.

In spite of the efficiency of the Louvain method, processing of a graph with 100 millions of nodes still takes several hours (Blondel et al., 2008). At the same time, it is known that dramatic acceleration of many data processing algorithms can be achieved by using *randomization* (Granichin et al., 2015). For instance, the randomized version of the clustering algorithm from Clauset et al. (2004), offered by Ovelgönne et al. (2010), proves to be much faster than its deterministic counterpart. In this paper, we describe an improved randomized version of the Louvain method, show its efficiency in comparison with the original method, and present test results for different input data.

The paper is organized as follows: Section 2 gives the required information about graphs, communities, modularity and the Louvain method. In Section 3 we present our randomized algorithm. Section 4 contains test results and comparison between the proposed algorithm and existing methods. The conclusion and plans for future work are in Section 5.

2. COMMUNITIES IN GRAPH

Consider a graph $G = (V, E)$ where $V \neq \emptyset$ is a set of vertices and $E \neq \emptyset$ is a set of edges. Let n and m be the number of elements of V and E respectively. A is an adjacency matrix where A_{ij} indicates the weight of the edge between nodes i and j, or 0 if there is no edge between them.

Communities (*groups* or *clusters*) are the sets of nodes $C = \{C_1, \ldots, C_k\}$ such that $\bigcup_{i=1}^{k} C_i = V$ and $\forall i, j \in 1, \ldots, k,\ i \neq j \quad C_i \cap C_j = \emptyset$, and where C is called a *clustering* of G. Communities C_i and C_j are adjacent to each other if $\exists i \in C_i, j \in C_j$ such that $A_{ij} \neq 0$.

A graph G has a community structure if the vertices can be easily divided into groups such that there is a higher density of edges within groups than between them.

Modularity

Modularity is a scalar value between -1 and 1 that measures the quality of clustering in the sense that there are many edges within communities and only few between them (Newman and Girvan, 2004). It is defined as

$$Q(G, C) = \sum_{i \in 1, \ldots, k} \left(e_{ii} - a_i^2\right) \quad , \tag{1}$$

where e is a $k \times k$ symmetric matrix whose elements e_{ij} are the fractions of all edges in the network that link vertices in C_i to vertices in C_j, and $a_i = \sum_{j \in 1, \ldots, k} e_{ij}$.

For a weighted graph it can be calculated by

$$Q = \frac{1}{2m} \sum_{i,j \in 1, \ldots, n} \left(A_{ij} - \frac{d_i d_j}{2m}\right) \delta(C_i, C_j) \quad , \tag{2}$$

where $d_i = \sum_{j \in 1, \ldots, n} A_{ij}$ is the sum of the weights of all edges attached to the node i, C_i is the community to which node i is assigned, δ-function $\delta(u, v)$ is 1 if $u = v$ and 0 otherwise, and $m = \frac{1}{2} \sum_{i,j \in 1, \ldots, n} A_{ij}$ (Newman, 2004a).

The Louvain Method

In 2008 Blondel et al. proposed one of the best known efficient algorithms of modularity maximization which is able to process a large sparse set of data in near linear time (Blondel et al., 2008; Martelot and Hankin, 2013), so-called Louvain Method.

The method's main idea is described by two phases:

(1) Local maximization of modularity by moving each node to neighbours' communities.
(2) Aggregate all of the nodes of the same community and build a new graph whose nodes are the communities from the previous phase.

The first phase stops when a local maximum is reached. The gain of modularity ΔQ derived from moving an isolated node i into a community C_j is

$$\begin{aligned} \Delta Q = &\left[\frac{\sum_{C_j, C_j} + \sum_{i, C_j}}{2m} - \left(\frac{\sum_{C_j} + \sum_i}{2m}\right)^2\right] \\ &- \left[\frac{\sum_{C_j, C_j}}{2m} - \left(\frac{\sum_{C_j}}{2m}\right)^2 - \left(\frac{\sum_i}{2m}\right)^2\right] \\ &= \frac{\sum_{i, C_j}}{2m} - \frac{\sum_{C_j} \sum_i}{2m^2} \quad , \end{aligned} \tag{3}$$

where $\sum_{C_j, C_j}$ is the sum of the weights of the edges inside C_j, $\sum_{C_j}$ is the sum of the weights of the edges incident to nodes in C_j, $\sum_i$ is the sum of the weights of the edges incident to node i, $\sum_{i, C_j}$ is the sum of the weights of the edges from node i to nodes in C_j, and m is the sum of the weights of all the edges in the network.

Initially each node in the network is assigned to its own community. Then, (1) for each node i we take the neighbouring communities C_j and compute the change in modularity $\Delta Q_{i,C_j}$ that would take place after moving the node i to the community C_j. If $\max_{C_j} \Delta Q_{i,C_j} > 0$ then we place the node i to the community on which the maximum is achieved. It repeats until a local maximum of modularity is attained. After that, (2) we build a new graph whose nodes are the communities C_j $\forall j$ and go to the first phase. These phases are repeated sequentially until no further improvement can be done.

For greater clarity, the steps of this algorithm are shown in Figure 1.

3. MAIN RESULT

The described algorithm works quite efficiently on a large amount of data but it takes a pretty long time on a graphs with billions of edges (Blondel et al., 2008).

In this paper we propose an improved version of Louvain Method based on the approach, applied in Randomized Greedy algorithm (Ovelgönne et al., 2010): at each iteration we take into account only random set of neighbours, what helps us to reduce computation time. Thus, the algorithm is (see Algorithm 1):

Algorithm 1 The proposed method

Input: $G = (V, E)$;
Output: clustering of G;
1: $k = 0$, $G^0 = G$;
2: **loop**
3: make a simple clustering C^k of G^k such that $C_i^k = \{i\}$;
4: **repeat** ▷ Phase 1
5: **for** node $i \in G^k$ **do**
6: remove the node i from its community C_i^k;
7: C_N = random set of neighbour communities of node i;
8: $C_j^k = \text{argmax}_{C_{j'}^k \in C_N} \Delta Q(i, C_{j'}^k)$;
9: **if** $\Delta Q(C_j^k, i) > 0$ **then**
10: add the node i to the community C_j^k;
11: **else**
12: leave the node i in the community C_i^k;
13: **end if**
14: **end for**
15: **until** no further improvement can be achieved
16: build a new graph G^{k+1} whose nodes are the communities of C^k; ▷ Phase 2
17: **if** $G^{k+1} = G^k$ **then**
18: make a clustering C_{final} of G;
19: **return** C_{final};
20: **end if**
21: $k = k + 1$;
22: **end loop**

This approach reduces the computation time, especially on large networks, and still provides a high coefficient of modularity.

In the next section we compare the original algorithm and the proposed randomized version. Also, we describe the dependence of the computation time and the resulting modularity from the number of considered neighbours.

4. COMPARISON OF ALGORITHMS

In this section we compare the Louvain method, Randomized Greedy (Ovelgönne et al., 2010) algorithm (with $k = 9$) and our improved method. For our algorithm, we consider the following cases which depend on the maximum number of considered neighbours k:

- $k = 75\%$ – the number of considered neighbours on each iteration is 75% of the total number of neighbours (denoted as $our_{k=75\%}$);
- $k = 50\%$ – the number of considered neighbours on each iteration is 50% of the total number of neighbours (denoted as $our_{k=50\%}$);
- $k = 25\%$ – the number of considered neighbours on each iteration is 25% of the total number of neighbours (denoted as $our_{k=25\%}$).

The testing was performed on a computer running Ubuntu 15.10 with Intel Core i5-5200U CPU (2.20GHz) and 16GB of RAM. To assess the quality and the computation time of these algorithms we use test graphs from *10th DIMACS Implementation Challenge - Graph Partitioning and Graph Clustering* which are located on the `http://www.cc.gatech.edu/dimacs10/archive/clustering.shtml`:

- karate.graph – Zachary's karate club: social network of friendships between 34 members of a karate club at a US university in the 1970s (n=34, m=78);
- as-22july06.graph – Internet: a symmetrized snapshot of the structure of the Internet at the level of autonomous systems, reconstructed from BGP tables posted by the University of Oregon Route Views Project (n=22963, m=48436);
- cnr-2000.graph – A very small crawl of the Italian CNR domain (n=325557, m=2738969);
- eu-2005.graph – A small crawl of the .eu domain (n=862664, m=16138468);
- in-2004.graph – A small crawl of the .in domain performed for the Nagaoka University of Technology (n=1382908, m=13591473);
- road_central.graph – A graph of roads (n=14081816, m=16933413);
- uk-2002.graph – This graph has been obtained from a 2002 crawl of the .uk domain performed by UbiCrawler. (n=18520486, m=261787258);
- road_usa.graph – A graph of roads (n=23947347, m=28854312);
- uk-2007-05.graph – This graph is a time-aware graph generated by combining twelve monthly snapshot of the .uk domain collected for the DELIS project. (n=105896555, m=3301876564).

Table 1 shows the approximate time work for different algorithms on test graphs, where *OOM* means *Out of memory.*

In Table 2 the average modularity for different algorithms on different graphs are presented.

According to the tables, our algorithm is much faster that the Louvain method and Randomized Greedy algorithm,

Fig. 1. Visialization of the steps of Louvain Method. Each pass consist of 2 phases. First phase: improve modularity by moving the nodes between communities. Second phase: aggregate the found communities in order to build a new network. Figure is reproduced from (Blondel et al., 2008).

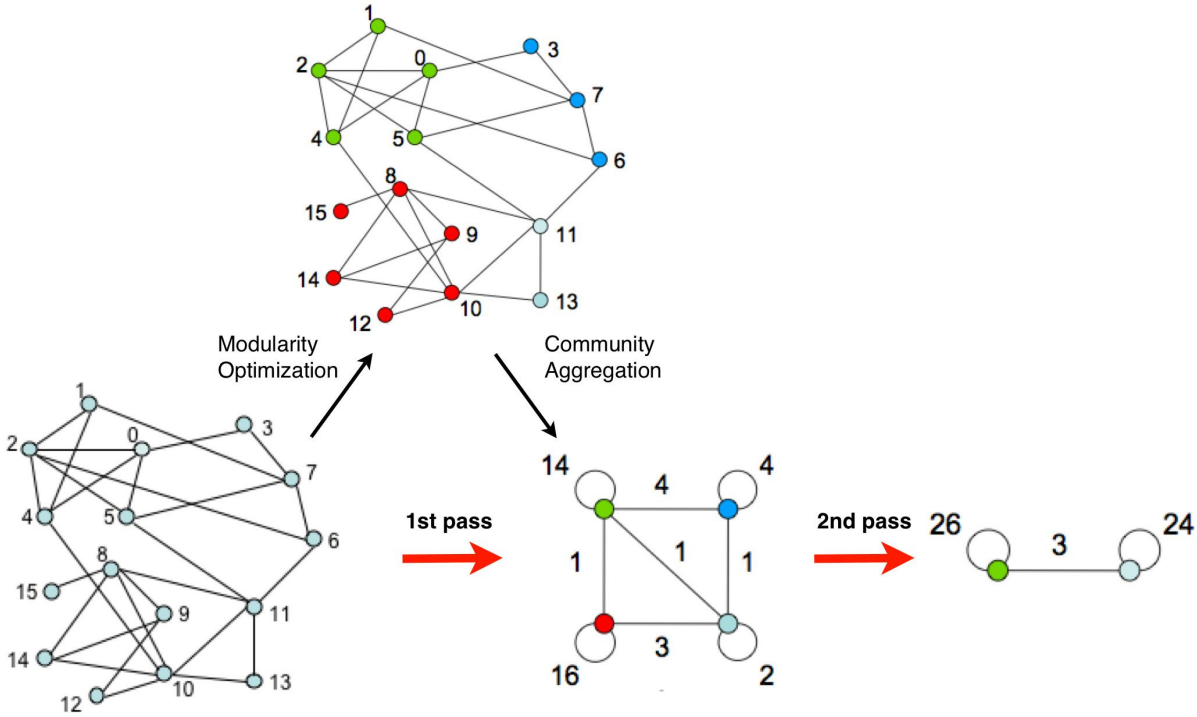

Table 1. The running time of *Louvain* method, Randomized Greedy (*RG*) and our randomized algorithm, seconds

	Louvain	$RG_{k=9}$	$our_{k=75\%}$	$our_{k=50\%}$	$our_{k=25\%}$
karate	0.000	0.000	0.000	0.000	0.000
as-22july06	0.036	0.072	0.028	0.036	0.044
cnr-2000	4.594	4.040	0.792	0.768	0.932
eu-2005	14.106	24.676	3.844	4.448	4.748
in-2004	26.646	21.396	3.156	3.516	4.368
road_central	123.028	114.824	36.612	36.016	44.392
uk-2002	433.468	OOM	70.976	71.256	70.880
road_usa	183.516	196.072	53.036	48.552	49.852
uk-2007-05	OOM	OOM	OOM	OOM	OOM

Table 2. Average modularity for *Louvain* method, Randomized Greedy (*RG*) and our randomized algorithm

	Louvain	$RG_{k=9}$	$our_{k=75\%}$	$our_{k=50\%}$	$our_{k=25\%}$
karate	0.41452	0.39423	0.35528	0.36037	-0.04980
as-22july06	0.66230	0.64839	0.61879	0.59751	0.48388
cnr-2000	0.91276	0.91051	0.91073	0.90602	0.88533
eu-2005	0.93822	0.92746	0.92280	0.89709	0.85685
in-2004	0.98020	0.96735	0.97707	0.97012	0.93383
road_central	0.99738	0.99723	0.99509	0.99205	0.98569
uk-2002	0.98973	OOM	0.94453	0.93721	0.94389
road_usa	0.99804	0.99791	0.99623	0.99370	0.99382
uk-2007-05	OOM	OOM	OOM	OOM	OOM

and it produces the partition with similar modularity. Thus, due to a slight deterioration of the quality of clustering, the described method can process large amount of data in a short time.

5. CONCLUSION AND FUTURE WORK

In this paper, we examine the Louvain Method for graph clustering and propose its randomized version (Algorithm 1). The suggested algorithm is capable to handle large-scale graphs much faster than the deterministic Louvain method without deterioration of the resulting clustering, measured by the modularity function.

The algorithm proposed in this paper can be further optimized by using the approach applied in Shiokawa et al. (2013). We are also working on the distributed version of the proposed algorithm and its comparison with other distributed methods for graph clustering, e.g. Wickramaarachchi et al. (2014).

REFERENCES

Albert, R. and Barabási, A.L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1), 47–97.

Blondel, V., Guillaume, J.L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), 10008.

Clauset, A., Newman, M., and Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(6), 066111.

Deco, G. and Corbetta, M. (2011). The dynamical balance of the brain at rest. *The Neuroscientist*, 17(1), 107–123.

Duch, J. and Arenas, A. (2005). Community detection in complex networks using extremal optimization. *Physical Review E*, 72(2), 027104.

Flake, G., Lawrence, S., Giles, C., and Coetzee, F. (2002). Self-organization and identification of web communities. *Computer*, 35(3), 66–71.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486, 75–174.

Girvan, M. and Newman, M. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99, 7821–7826.

Granichin, O., Volkovich, Z., and Toledano-Kitai, D. (2015). *Randomized Algorithms in Automatic Control and Data Mining*, volume 67 of *Intelligent Systems Reference Library*. Springer-Verlag Berlin Heidelberg.

Kleinberg, J. and Lawrence, S. (2001). The structure of the web. *Science*, 294(5548), 1849–1850.

Lancichinetti, A. and Fortunato, S. (2009). Community detection algorithms: A comparative analysis. *Physical Review E*, 80, 056117.

Lim, Y., Kang, U., and Faloutsos, C. (2014). Slashburn: Graph compression and mining beyond caveman communities. *IEEE Transactions on Knowledge and Data Engineering*, 26(12), 3077–3089.

Lu, Z., Sun, X., Wen, Y., Cao, G., and Porta, T. (2015). Algorithms and applications for community detection in weighted networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(11), 2916–2926.

Malliaros, F. and Vazirgiannis, M. (2013). Clustering and community detection in directed networks: A survey. *Physics Reports*, 533, 95–142.

Martelot, E. and Hankin, C. (2013). Fast multi-scale detection of relevant communities in large-scale networks. *Computer Journal*, 56(9), 1136.

Newman, M. (2004a). Analysis of weighted networks. *Physical Review E*, 70(5), 056131.

Newman, M. (2004b). Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6), 066133.

Newman, M. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23), 8577–8582.

Newman, M. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2), 026113.

Ovelgönne, M., Geyer-Schulz, A., and Stein, M. (2010). Randomized greedy modularity optimization for group detection in huge social networks. In *Proceedings of the fourth SNA-KDD Workshop, KDD 2010, July*, volume 25, 1–9.

Pons, P. and Latapy, M. (2006). Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications*, 10, 191–218.

Porter, M., Onnela, J.P., and Mucha, P. (2009). Communities in networks. *Notices of the American Mathematical Society*, 56(9), 1082–1097.

Ravasz, E., Somera, A., Mongru, D., Oltvai, Z., and Barabási, A.L. (2002). Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586), 1551–1555.

Rice, S. (1927). The identification of blocs in small political bodies. *American Political Science Review*, 21(03), 619–627.

Rosvall, M. and Bergstrom, C. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4), 1118–1123.

Schuetz, P. and Caflisch, A. (2008). Multistep greedy algorithm identifies community structure in real-world and computer-generated networks. *Physical Review E*, 78(2), 026112.

Shiokawa, H., Fujiwara, Y., and Onizuka, M. (2013). Fast algorithm for modularity-based graph clustering. In *Proceedings of 27th AAAI Conference on Artificial Intelligence*, 1170–1176.

Wakita, K. and Tsurumi, T. (2007). Finding community structure in mega-scale social networks. In *Proceedings of the 16th international conference on World Wide Web*, 153.

Wickramaarachchi, C., Frincu, M., Small, P., and Prasanna, V. (2014). Fast parallel algorithm for unfolding of communities in large graphs. In *High Performance Extreme Computing Conference (HPEC), 2014 IEEE*, 1–6.