A Comparative Study of Modularity-Based Community Detection Methods for Online Social Networks

Arzum Karataş and Serap Şahin

Izmir Institute of Technology, Izmir, Turkey
{arzumkaratas,serapsahin}@iyte.edu.tr

Abstract. Digital data represent our daily activities and tendencies. One of its main source is Online Social Networks (OSN) such as Facebook, YouTube etc. OSN are generating continuously high volume of data and define a dynamic virtual environment. This environment is mostly represented by graphs. Analysis of OSN data (i.e.,extracting any kind of relations and tendencies) defines valuable information for economic, socio-cultural and politic decisions. Community detection is important to analyze and understand underlying structure and tendencies of OSNs. When this information can be analysed successfully, software engineering tools and decision support systems can produce more successful results for end users. In this study, we present a survey of selected outstanding modularity-based static community detection algorithms and do comparative analysis among them in terms of modularity, running time and accuracy. We use different real-world OSN test beds selected from SNAP dataset collection such as Facebook Ego network, Facebook Pages network (Facebook gemsec), LiveJournal, Orkut and YouTube networks.

Keywords: Social network, Social network analysis, Community detection, Modularity, OSN.

Çevrimiçi Sosyal Ağlar İçin Modülerite Tabanlı Topluluk Algılama Yöntemlerinin Karşılaştırmalı Bir Çalışması

¿

Özet Dijital veriler günlük aktivitelerimizi ve eğilimlerimizi temsil eder. Bu verilerin ana kaynaklarından biri Facebook, YouTube vb. gibi çevrimiçi sosyal ağlardır (OSN). Sosyal ağlar sürekli olarak yüksek hacimli veriler üretir ve dinamik bir sanal ortam oluşturur. Bu ortam çoğunlukla çizgelerle temsil edilir. OSN verilerinin analizi (yani, her türlü ilişki ve eğilimin çıkarılması) ekonomik, sosyo-kültürel ve politik kararlar için değerli bilgilerin elde edilmesine katkıda bulunur. Topluluk algılama,

OSN'lerin altta yatan yapısını ve eğilimlerini analiz etmek ve anlamak için önemlidir. Bu veriler sağlıklı bir şekilde analiz edilebilir hale geldiğinde, yazılım mühendisliği araçları ve karar destek sistemleri son kullanıcı için daha başarılı sonuçlar üretebilir. Bu çalışmada, modülerliği, hızı ve doğruluğu açısından seçkin modülerite temelli statik topluluk algılama yöntemlerinin kısa bir araştırmasını sunduk ve aralarında karşılaştırmalar yaptık. Biz bu çalışmada Facebook Ego, Facebook Pages(gemsec), LiveJournal, Orkut ve YouTube ağları gibi SNAP veri seti koleksiyonundan gerçek dünya çevirim içi ağ test veri setlerini kullandık.

Anahtar Kelimeler: Sosyal ağ, Sosyal ağ analizi, Topluluk bulma, modularite, Çevrim içi sosyal ağlar.

1 Introduction

With proliferation of information technologies, we produce more digital data that mainly spring from our daily activities or tendencies. Huge amount of people communicates with each other, express their feelings, share daily routines and even personal things on Online Social Networks (OSNs) such as Twitter, Facebook, LinkedIn, YouTube etc. OSNs are so popular as a means of communication, advertisements and dynamic big data source. Individuals in OSNs form a relationship structure via connections of individuals and/or entities. Information is disseminated via those connections on the relationship structure as well.

Community detection reveals communities (i.e., set of individuals or entities heavily sharing common affiliations) on OSNs. It becomes an important field of Social Network Analysis (SNA), because it reveals the underlying structure of the OSN, helps analyzing people's view and public opinion on a topic, and examine information diffusion. Therefore, community detection helps gathering valuable information for economic, socio-cultural and politic decision making.

Community detection can be applied on static or dynamic networks [1] and social graphs may be directed/undirected or weighted/unweighted or multiple-edges. In this study, we only focus community detection on static networks and undirected, unweighted and single edged real world OSNs.

Community detection algorithms in social networks are reviewed by some other researchers. Wang [2] et al. make a depth benchmark between ten community detection algorithms such as CNM, LPA etc. within a procedure oriented framework. However, they do not regard famous modularity-based algorithms like Louvain, SLM etc. Emmons et al. [3] examine relationship between cluster quality metrics (e.g., modularity, conductance and coverage) and information recovery metrics (e.g.,NMI and ARI).However, we only focused on famous modularity-based algorithms. Additionally, the main contributions of this paper are to (i) compare performance of five outstanding static community detection algorithms on real life OSN datasets in terms of their modularity values, running time and accuracy, (ii) introduce existing static community detection methods and share their advantages and drawbacks and (iii) point possible research avenues for researchers on static community detection area for social networks.

Even if there are many category of community detection algorithms, we select algorithms based on modularity optimization because of four basic reasons. First, they are so prevalent. Second, they are easy to implement. Third, they provides low running time relatively. Finally, they are good for systems that contain big data like OSN.

The rest of the paper is organized as follows. In Section II, we give preliminary information about community, community detection, modularity metric and brief information about mainstream algorithms based on modularity optimization. In Section III, we introduce our experimental setup and procedure, then we discuss the experimental results. In Section IV, we close our paper by giving research avenues for community detection on social networks and concluding thoughts.

2 Mainstream Algorithms for Modularity Based Community Detection

2.1 Concept Definitions

Real world networks like social networks present an intrinsic community structure. The term 'community' does not have a universal definition but its definition depends on the context. However, it is widely accepted informal definition of community in SNA is that a subset of individuals heavily connected inside (i.e., have more common properties), rather than the rest of the network.

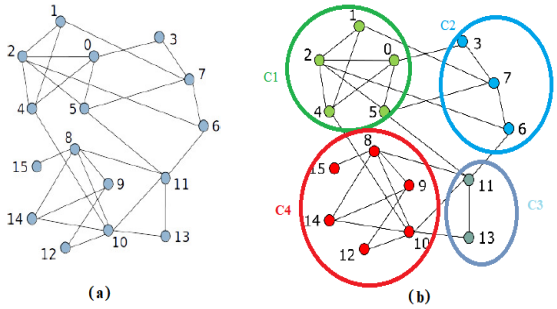
For a given network, represented by a graph G = V, E where V is the set of nodes and E the set of edges, the community detection problem consists of finding a partition of the nodes in V of the form C = C_1, . . ., C_k such that each C_i, 1 ≤ i ≤ k exhibits the community structure that presents groups of nodes so called communities [4].

Community detection in social networks addresses graph partitioning problem by dividing a network into interested attributes such as friendship relation, common geo-location, common interests etc. In Figure 1.(a)., a social network covering fourteen individuals (represented by nodes) is seen and detected four communities as c1, c2, c3 and c4 (indicated with rings) via a community detection algorithm is shown in Figure 1.(b). [5].

2.2 Static Community Detection Approaches

In social network domain, there are two broad community types as overlapping and non-overlapping (disjoint) communities. In this study, we focus on the problem of non-overlapping community detection, which aims finding a community structure that any individual can be member of only one community.

In social networks as in the case of the other real-world networks in other domains, exact solution of community detection problem can be NP-hard [6] due to the combinatorial nature of selection of the community members. Therefore, heuristic algorithms and approximation-based solutions become handy. It is vital



Şekil 1. (a) A sample social network (b) Detected Communities on the network [5]

to emphasize that identification of structural communities is computationally eligible for sparse networks like OSNs. There are many proposed algorithms for static community detection in the literature. Moreover, they can be categorized into fundamental categories as below:

Algorithms based on Partitional Clustering partition the network into predefined number of communities to optimize a given cost function based on distances. Most popular ones are k-means and its extensions [7] [8].They are usually easy to implement, but they need to specification of the number of communities in advance.

Spectral clustering algorithms [9] basically perform dimensionality reduction of the network before clustering in fewer dimension by using eigenvectors of matrices of network itself instead of other matrices derived from the network. They yield good results but not computationally efficient.

Algorithms based on Statistical inference [10] attempt to find a generative model from given network data for encoding existing community structure. They can work both overlapping and non-overlapping communities, and dynamic community detection as well. However, selection of models and high time complexity are drawbacks for them.

Algorithms based on Random walks [11] use the idea that random walks more likely to stay inside same communities because communities are densely connected inside. When doing a short random walk, probability of starting and ending individuals being in the same community is higher than being in other communities. They are adaptable for dynamic community detection as well. The limitation for them is that they need other clustering algorithms such as hierarchical clustering to work.

Algorithms based on Label propagation [12] are semi-supervised machine learning algorithms. They do not require either prior information about network structure or an objective function to optimize. They start with a set of individuals that each has a distinct community label. Those labels are propagated by largest numbers of neighbors of unlabeled (not assigned into a community yet) individuals at each step. They provide linear running time complexity; ho-

wever, they suffer from poor stability (e.g., they can produce quite a change community structure even if they work on same dataset more than once.)

Algorithms based on Modularity Optimization [7] [13] leverage modularity metric as determiner of quality of network partitioning. Those algorithms are based on approximation methods such as simulated annealing, greedy algorithms or other optimization methods balancing between speed and accuracy. They mostly suffer from resolution limit of modularity [6], which means small communities related to inherent edge numbers of the network cannot be detected. However, they have (nearly) logarithmic running time complexity and are easy to implement. Detailed information is given below.

2.3 Algorithms based on Modularity Optimization

Modularity(**Q**) is a metric that evaluates how a partition (or group or community) is modular which is distinguished by high number of intra-community connections with respect to expected inter-community connections. The most common modularity formula (proposed by Newman- Girvan) in community detection can be formulated as in Equation (1) [13] for unweighted and undirected graphs as in our study.

$$Q = \frac{1}{2m} \sum\nolimits_{i,j} [A_{ij} - \frac{d(i)d(j)}{2m}] * \sigma_{ij} \tag{1}$$

where A_{ij} is the adjacency matrix, m is the number of edges in the graph, d(i) and d(j) are the degrees of node i and node j respectively. σ is the function that returns 1 if both node i and j in the same community, else returns 0. Modularity value lies between -1 and 1. Higher modularity values implies strong community structure. Although many algorithms [5] [14] [15] [16] [17] [18] that aims to optimize modularity score, we select the most outstanding algorithms below.

Louvain: Blondel et al. [5] proposed Louvain algorithm that uses modularity function to optimize. It uses a greedy local approach and runs a local moving heuristic to obtain an improved community structure. The local moving heuristic follows the idea: repeatedly move individual nodes from one community to another neighbor community in such a way that each node movement results in a modularity increase.

Louvain contains two main phases in each iteration: modularity optimization via local moving heuristic and community aggregation. In the first phase, it starts with regarding each node in the network as a community; so initially each community is a singleton. Then, it uses the local moving heuristic to obtain an improved community structure by moving individual nodes from one community to another neighbor communities until no further increase in modularity can be achieved. In the second phase, it groups all the nodes belong to same community (e.g. merge the nodes) and construct a network where the nodes are the communities from the previous phase (e.g., construct a reduced network). It continues recursing until having only one community.

Louvain-gen: It is a generalized version of Louvain algorithm so that it can adapt other linear modularity functions as well. It is proposed by Campigotto

et al [15]. It can work both an unweighted network and a weighted network. It implements different modularity functions. It takes the dataset and convert it into a graph. Then, it computes communities with a specified quality function and displays hierarchical tree. However, in this study, we run it with modularity function in Equation (1).

Smart Local Moving (SLM): It is proposed by Waltman et al [16]. It is evolved from Louvain algorithm. While Louvain algorithm runs a smart local moving heuristic on the network and then build a reduced network, SLM algorithm changes the way to build the reduced network.SLM algorithm initially assigns each node to a different community as singleton communities. For each community, a subnetwork is built from copies of each community. Later, it runs the smart local moving heuristic on each subnetwork. After a community structure is obtained for each subnetwork, it builds the reduced network where the nodes are the communities detected in subnetworks. Later, it assigns the nodes in the reduced network to communities so that for each subnetwork, there is one community in the network. It starts the all process again by using reduced network instead of original one until a network is obtained that cannot be reduced more.

Combo: It is proposed by Sobolevsky and Campari [17] and it is an optimization algorithm for community detection that can deal with various objective functions. However, it outperforms the algorithms they compared with only modularity optimization.

Community detection algorithms adopt a search strategy that follows one of three ways for revealing community structure: merging two communities, splitting a community into two or movement of a node between two communities. However, Combo regards all of three strategies. It takes all nodes in the graph, initial communities (by default initially all nodes in one community) and the number of maximum communities (infinity by default). For each community pair regarded as source and destination, it calculates best gain from moving nodes from source community to destination community and store best partitions. Then, it performs the movements and update gains for changed communities. It uses Kernighan-Lin shifts [19] when it recalculates the gains.Since the algorithm regard all search strategies (merge, split and node movement), memory availability is a bottleneck for it. Its runtime upper bound is $O(N^2 \log(C))$, where N is the number of nodes in the graph and C is the number of communities in the network.

Complex Network Cluster Detection(Conclude): It is proposed by De Meo et al [18]. It (re)weights the edges in the network via k-path edge centrality (e.g., random, non-backtracking walks of finite length to compute the importance of each edge in keeping network connected). Those centralities allow nodes mapping on a Euclidean space. Then, Conclude calculates the distance between each pair of connected nodes on the space. Finally, it uses those distances used to partition the network into clusters via Louvain [5] Algorithm. The upper bound of running time of Conclude is linear in terms of the edge numbers $O(|E|)$, where E is the number of edges in the network.

3 Experiments and Results

3.1 Data Description

In this section, we validate five algorithms on five real-world networks from Stanford's Large Network Dataset Collection(SNAP) at snap.stanford.edu/data/: Facebook ego network, Facebook Pages (gemsec) network, YouTube network , Orkut network and LiveJournal network . We select datasets of those networks because Youtube, LiveJournal and Orkut networks contain ground-truth community information and the ground-truth communities for Facebook ego and gemsec networks can be built manually.

- **The Facebook Ego network dataset** contains friendship lists, user profiles (node features) and ego networks (e.g., personal networks) of users who are volunteers. This dataset is an anonymized dataset and it contains ground-truth community list as well. We accepted their ground-truth communities as our own ground-truth communities. We only use combined friendship list in the dataset to build an undirected and unweighted graph where each node represents a member and an edge represent a relationship among members. We collect ground-truth communities from the dataset manually.
- **The Facebook Pages (gemsec) network dataset** contains blue verified Facebook page networks of eight distinct type of pages such as government, new sites, athletes, public figures, TV shows, politician, artist and company. There is a separate .csv file for each page type. First, we create a new .txt file that includes all information on all type of pages. By using this file, we build an undirected and unweighted graph where nodes represent page and edges repre-sent common likes between the pages. We collect ground-truth communities from the dataset manually.
- In **YouTube**, users can form friendship and create user groups that the other members can join. Each connected component in a group as regarded a ground-truth community. We build an undirected and unweighted graph where nodes represent users and edges represent friendship among the users.
- In **Orkut OSN**, users can form friendship and create user groups that the other members can join. The user-defined groups are regarded as ground-truth communities. We build an undirected and unweighted graph where nodes represent users and edges represent friendship among the users.
- **LiveJournal** is a free online blogging site. Each user can build friendship and form a group that other users can join. The LiveJournal dataset contains LiveJournal friendship network and ground-truth communities are considered as user-defined groups. We build an undirected and unweighted graph where nodes represent users and edges represent friendship among the users.

Table 1 contains the descriptions of datasets for testing the community detection algorithms we select. The first column contains the networks. The second and third columns contain the number of nodes and edges respectively per network. Last column contains the number of ground-truth communities per network as well.

Tablo 1. Dataset descriptions in overall

Networks	Node Count	Edge Count	Community Count
Facebook Ego	4,039	88,234	10
Facebook Gemsec	134,833	1,380,293	8
Youtube	1,134,890	2,987,624	8,385
LiveJournal	3,997,962	34,681,189	287,512
Orkut	3,072,441	117,185,083	6,288,363

3.2 Procedure

We first select the algorithms to compare as Louvain, Louvain-gen, SLM, Combo and Conclude. We select these modularity-based community detection algorithms in this work because they are preferable by big social networks (our domain) like Twitter, Facebook, YouTube etc. because of their (near) logarithmic time complexity and availability of implementation of these algorithms on web at cse.iitkgp.ac.in/resgrp/cnerg/permanence/, ludowaltman.nl/slm/, senseable.mit.edu/community_detection/and emilio.ferrara.name/code/conclude/.

We design the following experiment in the direction of our aims. First, we download either executable or source code of the algorithm and run them on a laptop with Core i7 2.30 GHz CPU and 8GB memory. Then, we select our testbeds as we mention just above part. We prepared the datasets so that they can feed the algorithms. For example, we add unit weights (e.g. 1) to the datasets so that we can run Louvain-gen algorithm. Additionally, we convert .csv files into .txt files for gemsec dataset to obtain community ground-truth. Additionally, we convert .txt files that contain network information into Pajek .net for feeding Combo algorithm via using a tool introduced in [20]. Later, we run the algorithms on each dataset and obtained community structure detected and running time per each algorithm.

Since our evaluation metric (e.g., F1-Score [21]) needs that every community to evaluate should have at least three members. Therefore, we modify the .txt files that include ground-truth communities and community structure information produced by the selected algorithms by eliminating the communities contain only one or two members inside.

3.3 Evaluation

It is possible to evaluate the quality of detected communities to use either internal measures (i.e., scoring functions such as conductance, triangle partition ratio etc.) or external measures (i.e., comparison with ground-truth communities) such as Normalized Mutual Information, Adjusted Rand Index etc. [22]. We use F1-score to assess the quality of detected communities because of twofolds: linear computational complexity in terms of community size and easy to interpret among the external measures.

F1-score is common score in binary classification, which is harmonic mean of precision (e.g., the proportion of positive identifications is correct) and recall

(e.g., the proportion of actual positives is identified correctly). It is stated as in Equation (2).

$$F1(C, C') = 2 * \frac{precision(C, C') * recall(C, C')}{precision(C, C') + recall(C, C')} \quad (2)$$

where C is the ground-truth community and C' is the predicted community, precision (C, C′) is |C∩C′ | / | C′ | and recall (C, C′) is | $C \cap C'$ | / | C |. The higher F1 score, the higher community partition quality.

For Facebook ego network as seen from Table 2, nearly same modularity values are produced by the all algorithms, but Louvain-Gen has the smallest time consumption with 0.1 second and Conclude has the highest time consumption with 14 minutes.

Tablo 2. Facebook Ego Network

Algorithms	Q	F1	Time
Louvain	0.83558	0.567	0.2 sec.
Louvain-gen	0.83494	0.162	0.1 sec.
SLM	0.83579	0.535	102 sec.
Combo	0.83587	0.355	20 sec.
Conclude	0.84695	0.230	14 mins.

For the gemsec network seen in Table 3, modularity values of Louvain, Louvain-gen and SLM are near. The highest F1 score is belonged to Louvain. Louvain-gen is the fastest algorithm for running this dataset with 2 seconds. However, Conclude is the slowest algorithm with 2.5 days. Another thing, Combo does not work for this network because of its 30000 node limits. "x" symbol indicates that it will not work. Therefore, it F1 score and time consumption cannot be calculated (e.g., NA- Non-Available). After this dataset, Combo and Conclude are out of order node and time limitation, respectively for the rest of the datasets.

Tablo 3. Facebook Gemsec Network

Algorithms	Q	F1	Time
Louvain	0.44873	0.109	11.2 sec.
Louvain-gen	0.40584	0.009	2 sec.
SLM	0.44896	0.099	28 sec.
Combo	x	NA	NA
Conclude	0.27633	0.001	2.5 days

For the YouTube network seen in Table 4, note that we assume the output of algorithms as NA working more than 2.5 days. All algorithms produce nearly same F1 scores and closely same modularity values. Louvain-gen performs best

in term of time with 20 seconds and SLM has the worst time consumption with 470 seconds among them. Louvain lies between them for all performance criteria.

Tablo 4. Youtube Network

Algorithms	Q	F1	Time
Louvain	0.72793	0.065	181 sec.
Louvain-gen	0.70970	0.066	20 sec.
SLM	0.73166	0.065	470 sec.
Combo	x	NA	NA
Conclude	NA	NA	NA

For the Orkut network seen in Table 5, the all three produces near modularity values and F1 scores, but Louvain-gen has the best time consumption with 97 seconds while SLM has the worst time consumption among them with nearly 34 minutes. Louvain lies between them for all performance criteria.

Tablo 5. Orkut Network

Algorithms	Q	F1	Time
Louvain	0.72729	0.092	500 sec.
Louvain-gen	0.71510	0.076	97 sec.
SLM	0.73106	0.096	≈34 mins.
Combo	x	NA	NA
Conclude	NA	NA	NA

Tablo 6. LiveJournal Network

Algorithms	Q	F1	Time
Louvain	0.79168	0.044	766 sec.
Louvain-gen	0.77533	0.044	146 sec.
SLM	0.79335	0.047	≈ 63 mins.
Combo	x	NA	NA
Conclude	NA	NA	NA

For the LiveJournal network seen in Table 6, the all nearly have same modularity value and F1 score. However, Louvain-gen beats them in terms of time consumption with 146 seconds whereas SLM has the worst time consumption among them with nearly 63 minutes.

4 Research Directions and Conclusion

In this work, we first provide the reader with an overview of existing static community detection methods according to their techniques used and give pros and

cons for each them. We realize points the below as open problems for community detection on OSN:

- **Stability of community detection (CD) algorithms:** Either existing unstable algorithms may be modified or a new one can be proposed so that with slight changes algorithm produce stable community structure on the dataset.
- **Scalability of CD algorithms:** It can gracefully response with growth of the OSN, and therefore they can deal with big data. Therefore, new graph algorithms or new data structures can be designed, or existing ones can be modified. Additionally, a new representation for OSN can be developed.
- **Refinement on Computational Complexity of more accurate algorithms:** Computational complexity of the algorithms like spectral clustering-based algorithms can be tried to decrease for more accurate result with graceful time requirement. Also, new optimization metrics can be defined.
- **Dynamicity:** As a next step of static CD algorithms, dynamicity of OSN should be regarded, and dynamically community detection and tracking algorithms should be contributed.
- **Prediction:** As final step, a researcher may develop some methods for prediction of future of communities in terms of either possible friendships or community events like growing, shrinking, death etc.

Additionally, in this study we select the algorithms using one type of modularity function, e.g., Newman-Girvan modularity. As future work, it is possible to research other linear modularity functions in community detection area.

We have present a comparative study of community detection algorithms based on modularity. We select five outstanding algorithm and tested them in our Experiment part via five-real world OSN networks and F1-score. Our comparison reveals that only Louvain, Louvain-Gen and SLM can detect communities in allowed time limit. We see that Combo can deal with only small datasets and this is not acceptable for OSN, also Louvain performs in one tenth time of it. Similarly, Conclude needs more time than 2.5 days for a million edges and it cannot beat Louvain in terms of both F1-score and time. Our final thought is that Louvain beats the other algorithms when we regard our three performance criteria (modularity value, F1-score and running time) together.

References

1. D. Greene, D. Doyle, and P. Cunningham:Tracking the evolution of communities in dynamic social networks. pp. 176-183.In Proocedings of International conference on Advances in social networks analysis and mining (ASONAM),pp. 176-183,IEEE, Odense,Denmark(2010)
2. Wang M., Wang C., Yu J. X. and Zhang, J.: Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework. In Proceedings of the VLDB Endowment, 8(10), 998-1009(2015).
3. Emmons S., Kobourov S., Gallant M. and Börner, K.: Analysis of network clustering algorithms and cluster quality metrics at scale. PloS one, 11(7), e0159161(2016).

4. S. Fortunato, and M. Barthelemy:Resolution limit in community detection.Proceedings of the National Academy of Sciences 104(5), 36-41(2007)
5. Blondel V. D., Guillaume J.L., Lambiotte R., and Lefebvre E.:Fast unfolding of communities in large networks.Journal of statistical mechanics: theory and experiment 2008(10), 10008(2008)
6. Fortunato S.: Community detection in graphs.Physics reports 486(3-5), 75-174(2010)
7. MacQueen J.:Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability 1(14), pp.281-297, Berkeley, California(1967)
8. Rattigan M. J.,Maier M.,and Jensen D.: Graph clustering with network structure indices.In Proceedings of the 24th international conference on Machine learning(ICML), pp. 783-790, ACM, Oregon, USA(2007)
9. Newman M. E.:Finding community structure in networks using the eigenvectors of matrices.Physical review E 74(3),036104(2006)
10. Newman M. E.:Models of the small world.Journal of Statistical Physics 101(3-4),819-841(2000)
11. Pons P., Latapy M.:Computing communities in large networks using random walks.In Proceedings of International symposium on computer and information sciences, pp. 284-293,Springer, Berlin, Heidelberg(2005).
12. Raghavan U. N., Albert R. and Kumara S.:Near linear time algorithm to detect community structures in large-scale networks.Physical review E 76(3), 036106(2007)
13. Newman M. E. and Girvan M.:Finding and evaluating community structure in networks. Physical review E 69(2), 026113(2004)
14. Clauset A., Newman M. E. and Moore C.: Finding community structure in very large networks. Physical review E 70(6), 066111(2004)
15. Campigotto R., Céspedes P. C. and Guillaume J. L.:A generalized and adaptive method for community detection.arXiv preprint arXiv:1406.2518, 2014
16. Waltman L. and Eck N. J.: A smart local moving algorithm for large-scale modularity-based community detection.The European Physical Journal B 86(11), 471(2013)
17. Sobolevsky S., Campari R.,Belyi A. and Ratti C.: General optimization technique for high-quality community detection in complex networks. Physical Review E 90(1), 012811(2014)
18. De Meo P., Ferrara E., Fiumara G. and Provetti A.Mixing local and global information for community detection in large networks. Journal of Computer and System Sciences 80(1), 72-87(2014)
19. Kernighan B. W.and Lin S.: An efficient heuristic procedure for partitioning graphs.The Bell System Technical Journal 49(2), 291-307(1970)
20. Pfeffer J., Mrvar A. and Batagelj V.: txt2pajek: Creating Pajek Files from Text Files. Technical Report, CMU-ISR-13, 2013
21. Rossetti G., Pappalardo L. and Rinzivillo S.:A novel approach to evaluate community de-tection algorithms on ground truth. Complex Networks VII,pp. 133-144, Springer, Cham, 2016.
22. Fortunato S. and Hric D.:Community detection in networks: A user guide. Physics Reports 659, 1-44(2016)