Social Network Analysis and Mining (2019) 9:23
https://doi.org/10.1007/s13278-019-0566-x

ORIGINAL ARTICLE

Community detection in large-scale social networks: state-of-the-art and future directions

Mehdi Azaouzi[1] · Delel Rhouma[1] · Lotfi Ben Romdhane[1]

Received: 10 July 2018 / Revised: 29 April 2019 / Accepted: 5 May 2019 / Published online: 18 May 2019

Abstract
Community detection is an important research area in social networks analysis where we are concerned with discovering the structure of the social network. Detecting communities is of great importance in sociology, biology and computer science, disciplines where systems are often represented as graphs. This problem is an NP-hard problem and not yet solved to a satisfactory level. This computational complexity is hampered by two major factors. The first factor is related to the huge size of nowadays social networks like Facebook and Twitter reaching billions of nodes. The second factor is related to the dynamic nature of social networks whose structure evolves over time. For this, community detection in social networks analysis is gaining increasing attention in the scientific community and a lot of research was done in this area. The main goal of this paper is to give a comprehensive survey of community detection algorithms in social graphs. For this, we provide a taxonomy of existing models based on the computational nature (either centralized or distributed) and thus in static and dynamic social networks. In addition, we provide a comprehensive overview of existing applications of community detection in social networks. Finally, we provide further research directions as well as some open challenges.

Keywords Social networks · Dynamic social network · Centralized community detection · Distributed community detection · Semantic

1 Introduction

We are daily weaving links of different natures with people. All these relations, considered collectively, constitute social networks, which have been long the subject of study by sociologists, behaviorists, economists, etc. Thus, Wasserman defined a social network as a set of actors interconnected via relationships created during the interaction (Wasserman and Faust 1994). In many contexts, social networks can be modeled by graphs in which each entity is represented by a node and interaction is represented by an edge (Fortunato 2010). These graphs, which we call complex networks can be encountered in the real world in disciplines such as computer science, sociology, physics, biology, etc. (for a complete overview see Costa et al. 2011). The advent of online social networks as well as the availability of information concerning not only the relations which exist between the actors, but also the data allowing to describe or characterize them, has led to a renewed interest in their analysis (Fortunato 2010). This analysis can predict the characteristics of the actors, the appearance of any links, the arrangements of diffusion in the network, etc. Although the graphs had no obvious structural property, they have common one that characterizes them independently of their specific content. For example, Milgram (1967) suggests that the average distance between two nodes is very low compared to the total size of the graph [two randomly selected people are connected on average by a chain of six relationships ("six degrees of separation")]. This is the phenomenon of "small-world" (Milgram 1967). Another property is that the number of neighbors of a vertex is often distributed in a heterogeneous law (Faloutsos et al. 1999) (usually well approximated by a power law) many nodes have few neighbors and some of them (acting as hubs) have many adjacent vertices (Albert

✉ Mehdi Azaouzi
mehdi.azaouzi@gmail.com

Delel Rhouma
rh.delel@gmail.com

Lotfi Ben Romdhane
lotfi.BenRomdhane@isitc.u-sousse.tn

[1] Modeling of Automated Reasoning Systems Research Laboratory LR17ES05, Higher Institute of Computer Science and Telecom, University of Sousse, Sousse, Tunisia

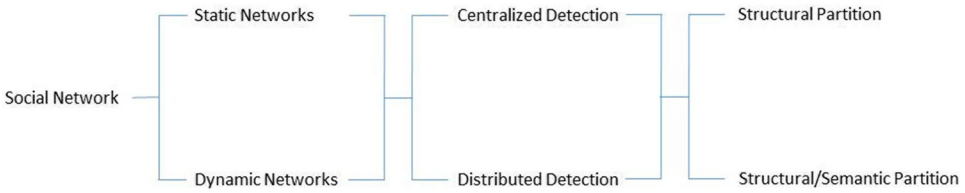

Fig. 1 Taxonomy of models for community detection in social networks

and Barabási 2002). The third property of the graphs is their low density. This low density is usually related to individual limits of each actor that cannot establish many relationships. Therefore, the average degree of the graph is very low contrary to clustering coefficient which is high (Albert and Barabási 2002). Indeed, two of your friends are more likely to know each other (friends of my friends, etc.) that two people chosen randomly from the network because the overall density is low. The difference in density between the global and local levels in the graphs is generally explained by the presence of groups of nodes strongly related to each other and loosely connected with the outside called "community" (Girvan and Newman 2002).

Identifying community allows us to obtain a macroscopic view of complex systems and it is a valuable tool to understand and analyze these systems. From a theoretical point of view, community detection is a NP-complete graph-partitioning problem (Fortunato 2010; Xie and Szymanski 2013; Ben Romdhane et al. 2013; Rhouma and Romdhane 2014).

Communities are an important feature of the network that can generally be defined as two categories: the structure based, a cluster of nodes that have more links with each other than other nodes of a graph, and the semantic based, a cluster of nodes with similar semantic context or a set of nodes sharing the same interests. Since the semantic communities are detected by both context and relationship of the nodes, the result could represent the cohesion of communities more efficiently. Characterizing both structures and semantic simultaneously is a challenging task because of the diverse characteristics of topological structures and semantic context.

The community detection constitutes a growing challenge, because of the heterogeneity of data and structures formed in them, and their size. When we add the dynamics in this wealth of data, the problem has turned even more complicated. These limitations motivated an increasing interest on distributed algorithms for large-scale network. These approaches could be roughly classified into two categories. The first and most studied one aims to distributed detection framework with shared-memory parallelism or "distributed computing". These parallelization strategies divide the social network into non-overlapping subnetworks that are treated in parallel. However, the high cost of shared memory multiprocessor systems and the high degree of data dependency, are seen as impediments for this strategy. In this direction, attempts were made to detect several communities in a distributed manner using an *undistributed* social graph. Here, we conduct a part of our survey on the state-of-the-art models for detecting parallel the community structure.

The problem of large size of current social networks is amplified by the dynamic changes of the networks over time, which is reflected by the addition and the disappearance of nodes, links and their contents posing several difficulties in the calculation. Community detection in dynamic networks involves the process of incorporating the community model of a previous timestamp, or snapshot of a network structure, into the detection of the next to improve the efficiency of detecting the new community structure.

The search for such communities attracted considerable attention in recent years, and many studies have been devoted to study community structure in graphs. These researchers are trying to cope with the huge dynamic amount of data and the joint exploitation of relational and semantic data.

1.1 Goals and contributions of the survey

The main goal of this survey is to present, organize, and analyze the existing models developed for determining the communities of the given large-scale network. The focus of the paper is:

1. To categorize and compare the theoretic community detection algorithms.
2. To bring together techniques related to characterize, identify, and extract communities with two strategies: centralized and distributed.
3. To analyze the strategy of detection according to the dynamic of networks: static or evolves over time.
4. To provide the metrics for detecting partition: the structural-based partition or the semantic and structural-based partition.

The paper is divided into four parts. We provide the taxonomy of models (both centralized and distributed) available in the literature for detecting the communities of the given social network (both static and dynamic) as follows in Fig. 1.

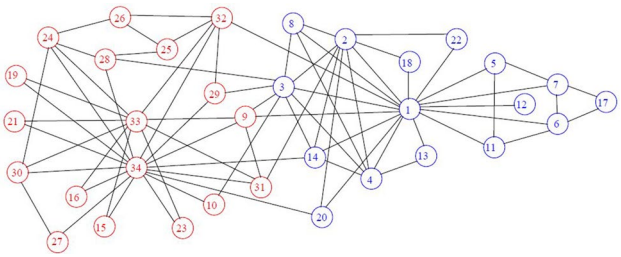

Fig. 2 Zachary's karate club with two disjoint communities

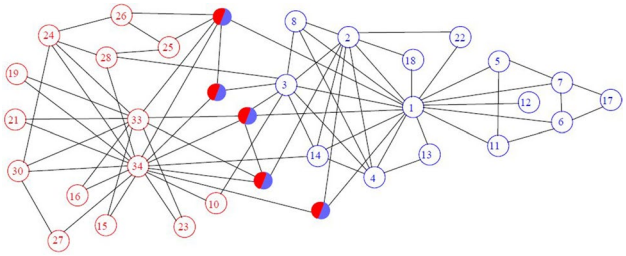

Fig. 3 Zachary's karate club with two overlapping communities

1.2 Structure of the survey

To summarize, the rest of the paper is organized as follows. In Sect. 2 we discuss the preliminaries pertained to community structure theory as well as the large-scale social networks concept. Section 3 discusses the taxonomy of *centralized* models for community detection in *static* social networks. The same strategy (centralized) for community detection in *dynamic* social networks is discussed in Sect. 4. The nature of partition: structural/structural and semantic, is described with the centralized community detection in these two types of networks. The *distributed* detection models in the static and dynamic networks are presented, respectively, in Sects. 5 and 6. A comparative analysis is presented in Sect. 7 based on the experiments conducted on real-world networks. The promising applications of community detection are presented in Sect. 8 to emphasize the potentiality of community detection. In Sect. 9, we discuss the future directions and open challenges in social networks using community detection. Then, we conclude this survey by providing our remarks in Sect. 10.

2 Preliminary material

This section is devoted to the terminology and definitions that will be used throughout this paper. A social network can be modeled as a graph $G(V, E)$, where V denotes the set of nodes or vertices, and $E \subseteq VXV$ denotes the set of edges (e.g., 34 nodes in Fig. 2). Every node (sample, labeled (content)), here, represents an actor and if two actors know each others, an edge (oriented, bidirectional, labeled (content)) is added between them.

A community C_i may be described as a group of vertices (i.e., a subgraph) that probably share common properties and/or play similar roles within a network (see Fortunato 2010). Formally, a community is a group of vertices having a high density of edges within them, and a lower density of edges between groups. A community structure (or clustering) is defined as a division $C = \{C_1, \ldots, C_k\}$ of the network in k subgraphs such that $V = \cup_k^{i=1} C_i$. Communities

Table 1 Table of notations

Notations	Descriptions
$G(V, E)$	A graph with a set of vertex V and a set of edge E.
n	The number of nodes of G, i.e., $\|V\|$
m	The number of edges of G, i.e., $\|E\|$
d	The degree of a node
N_c	Number of communities to detect
l	Number of levels in multilevel community detection
$N\,u$	Set of the neighbors of a node u
C_i	A community of the graph G

in real-world networks are of different kinds: disjoint or overlapping. The first approach (determinist) provides a disjoint (each node belongs to only one community) clusters of the graph, i.e., $\forall(i \neq j), C_i \bigcap C_j = \emptyset$. For example, students belonging to different disciplines in an institute. The second overlapping approach detects joint clusters (nodes participate in more than one cluster communities), i.e., $\forall(i \neq j), C_i \bigcap C_j \neq \emptyset$ (see Xie and Szymanski 2013). For example, a person having membership in different social groups on Facebook. Figures 2 and 3 illustrate an example of these two types of communities C_1 and C_2 for a Zachary's karate club network. In case of non-overlapping communities, each node belongs to only one community, while for overlapping communities, nodes can belong to multiple communities (e.g., nodes 9, 20, 29, 31 and 32 in Fig. 3 belongs to both C_1 and C_2).

Notations in Table 1 will be used throughout this article. Note that, we will use the terms "node" and "vertex", "edge" and "link", "cluster" and "community" and "group" interchangeably, as well as "network" and "graph".

The ill-posed definition of the community structure makes the evaluation framework complicated in the sense that there is no universally accepted metric for evaluating a community detection algorithm. There has been plenty of metrics proposed in the literature that are qualified to evaluate a community detection algorithm such as modularity, separability, cohesiveness, etc. Here is, an extensive review

of existing metrics for discovering community structure in Chakraborty et al. (2017).

2.1 Large-scale social networks

So far there is little consensus about what large networks are, because the term of "large" may drastically differ from one environment to another. Further, the term of large network depends on what you want to calculate. In order to create and manipulate it, it is enough if it fits into the memory. For example, a graph with one million vertices and ten million edges needs about 320 Mbytes. With the rapid increase of memory space, community detection in these large networks has become a key aspect of research field.

The small networks (some hundreds of vertices), if they are not dense, can be represented by a picture and analyzed by many algorithms. Till 1990 most networks were small, they were collected by researchers. The advances in IT allowed to create networks from the data already available in the computer(s) or by browsing on the Internet. This even recently introduced a new term in computer sciences: "Big data" and particularly in real-world networks as "large-scale" networks. Large networks, having thousands of vertices and lines, can be found in many different areas, e.g.: genealogies, flow graphs of programs, molecule, computer networks, transportation networks, social networks, intra-/inter-organizational networks, etc. One of the main large networks is social networks. As an example, Facebook has now more than 2.2 billion users every month and more than 1.4 billion people use it daily,[1] Twitter has currently about 330 million monthly active users,[2] and LinkedIn reached more than 260 million monthly active users.[3] If we consider networks of User Generated Content, in 2015 the number of photographs shared on Flickr more than ten billion photographs.[4] And in 2018 more than 5 billion YouTube videos have been shared.[5] A collection of large networks is available from Pajek's datasets.[6]

Observing the importance and significance of methods of communities' detection, with space and time constraints, to social network analysis, here we provide a comprehensive review of the available algorithms and studies for community detection in these large networks.

[1] https://newsroom.fb.com/news/2018/04/facebook-reports-first-quarter-2018-results/, Apr. 2019.

[2] https://blog.hootsuite.com/twitter-statistics/, Apr. 2019.

[3] https://www.omnicoreagency.com/linkedin-statistics/, Apr. 2019.

[4] http://blog.flickr.net/en/2015/05/07/flickr-unified-search/, Apr. 2019.

[5] https://www.omnicoreagency.com/youtube-statistics/, Apr. 2019.

[6] http://mrvar.fdv.uni-lj.si/pajek/, Apr. 2019.

3 Centralized community detection in static social networks

3.1 Centralized community detection based on structural partition

In this section, we aim to examine the trend and progress of centralized clustering algorithms to deal with large data taking into consideration only the structure of the network. One approach is to manage the detection level by level ("multilevel community detection"). Another is the graph sampling, which did not recently receive much attention ("reduction of the graph's size") (see Hu and Lau 2013).

3.1.1 Multilevel approach

The introduction of a new approach during 1990s, called "multilevel", provided a breakthrough in the resolution of large-scale classification problems (Chevalier and Safro 2009). This technique has been applied in several branches of science, such as the design of VLSI (Very Large-Scale Integration) (Chevalier and Safro 2009), the graph optimization (Safro et al. 2009) (especially graph partition by Barnard and Simon 1994; Hendrickson and Leland 1995; Karypis and Kumar 1998a) and several others (Chevalier and Safro 2009).

The major goal of the algorithms based on the concept of multilevel is to create a hierarchy of problems (Hendrickson and Leland 1995). In fact, the initial graph size is reduced level by level with the collapse of vertices and edges. This contraction ends with a reduced graph having a limited number of nodes and edges, which will be partitioned with a suitable clustering algorithm. The last step consists in making a projection in a sequential way along all levels until it reaches the initial graph. To improve the projection quality these approaches use refinement algorithms (Hendrickson and Leland 1995). The most representative algorithms are: METIS (1998a), Graclus (2007), Louvain (2008a), MLR-MCL (2009) and Nerstrand (2015).

The multilevel process consists of three stages: "coarsening", "initial partition" and "uncoarsening and refinement" (shown Fig. 4), and these will be detailed in the following paragraphs.

The steps of the multilevel approach are:

1. The phase of coarsening: The goal of this step is to convert the original graph, level by level, into smaller ones. The contraction starts with the initial graph G_0 then a series of small graphs $G_1;G_2 \ldots;G_l$ is generated such that $|V_i| > |V_{i+1}|$. The construction of G_{i+1} takes into account G_i and is as follows: The set of nodes in G_i is combined into supernodes in G_{i+1} to form the coarsened

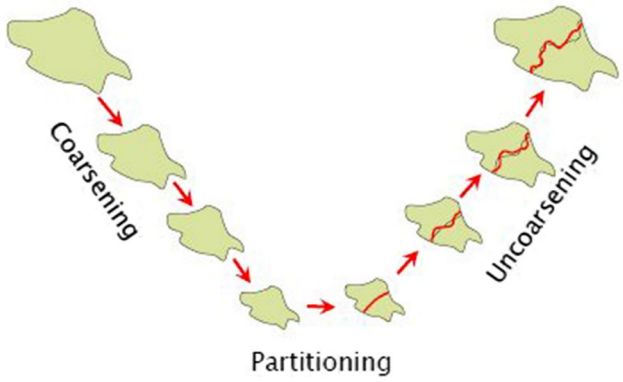

Fig. 4 The stages of multilevel partitioning

graph. Several algorithms have been developed, but the initial one is created by Hendrickson and Leland (1995) which is based on this idea: in the beginning, all vertices are unmarked. Then, a vertex is visited at random, and merged with one of its unmarked neighbors. The resulting vertex of merging is noted as a supernode. When a node is visited or contracted with another, it will be marked. If a vertex does not have unmarked neighbors, it will be matched by itself. This procedure is repeated until all vertices are marked or no more coarsening can be made.

2. The phase of initial partition: The contracted graph (obtained at several levels) will be partitioned directly. The initial algorithm of clustering used at this stadium varies according to different methods.
3. The phase of uncoarsening: This phase is the opposite of the first one. Indeed, the partition of G_i is deducted from G_{i+1}. This stage consists, first of all, in projecting a grouping of G_{i+1} in G_i (Whang et al. 2012). Then, it refines the grouping of G_i level by level to obtain the final classification of the original graph.

Several techniques have adopted the multilevel methodology to cope with the enormous size of data: Hendrickson and Leland (1995) proposed an algorithm (ML) inspired by the work of Barnard and Simon (1994). They use a multilevel approach to calculate eigenvectors during the spectral bisection in a recursive way. This raised an intense numerical problem in the transfer of eigenvectors between levels. To overcome this drawback, the authors of Hendrickson and Leland (1995) presented an algorithm that consists of three parts: the first is to build a sequence of coarsened graph by adding weights on the links and nodes. This is done with the use of the maximal matching (especially, the random matching (RM)). Then the last graph is partitioned in a spectral manner. Finally, this grouping is projected back (level by level) and is periodically enhanced with a local refinement algorithm (Kernighan and Lin 1970). The fundamental idea of KL is to maximize the gain of moving a node i form a cluster C_l to another C_k.

$$g^k(i) = \sum_{(i,j)\in E} \begin{cases} w_{ij}c_{kl}, & \text{if } P(j) = k \\ -w_{ij}c_{kl}, & \text{if } P(j) = l \\ w_{ij}(c_{lm} - c_{km}), & \text{if } P(j) = m, m \neq k, m \neq l \end{cases} \tag{1}$$

where $P(j)$ is the current set for vertex j, w_{ij} is the weight of the edge between vertices i and j, and c_{lk} is the symmetric inter-set cost metric for an edge between sets l and k. The main advantage of this algorithm is that the constraints on group sizes that depend only on the number of nodes and links are kept in the contracted weight graph. Therefore, good partition of the contracted graph corresponds to a correct partition of the original graph. The cost of the contraction is proportional to the number of edges of the graph. Although this algorithm has demonstrated an excellent performance in a variety of graphs, it requires many memories for storing the graphs during the coarsening phase. In fact, an aggressive coarsening strategy generating fewer intermediate graphs would be one way of reducing this problem (Hendrickson and Leland 1995).

The method of Mansour et al. (1993) that preceded the method of Hendrickson and Leland (1995) is driven by purely physical notions. In fact, it makes a sequence of "mapping" graph to reduce its size but it does not use any refinement algorithm (the final partition is a direct injection of coarsened graph).

The algorithm METIS by Karypis and Kumar (1998a) is the improvement of the work of Hendrickson and Leland (1995). It performs recursive coarsening, partitioning and uncoarsening separately. The particularity lies in the phases of contraction and refinement. Indeed, a new heuristic is used (hard edge heuristic) as well as a variation of the algorithm of refinement (variation of KL (Boundary KL (BKL)) which is faster). Concerning the partition phase, METIS tries to find balanced groups, i.e., communities having similar size. For this, it is coupled with different bisection algorithms: spectral bisection algorithm and others using edge cut minimizing strategy (see Ruan et al. 2015).

The local optimum of coarsened graph G_{i+1} does not necessarily lead to a local optimum of G_i. Thus, METIS uses BKL which inserts the gains for only the boundary vertices (vertices are only inserted into data structures as needed), to reach a local optimum on a finer graph (Karypis and Kumar 1998a). Indeed, this algorithm detects a good partition of graph in $\mathcal{O}(m)$ which is faster than algorithms of bisection.

The previously quoted methods provided groups of equal size, and Dhillon et al. (2007) remove this restriction. Indeed, it adopts weighted kernel k-means (a variant of kernel k-means appending a weight to each cluster) to perform both the partitioning and refinement phases. In this case,

Graclus guarantees that graph clustering is done by using a simple iterative algorithm without computing eigenvectors. Moreover, by focusing only on the boundary vertices, weighted kernel k-means can be faster with little loss in cluster quality. The time complexity of this method is $\mathcal{O}(kn)$ (Dhillon et al. 2007). The experimental results show that this algorithm surpasses the others regarding speed, use of memory and quality.

To cope with the two major limitations of Markov Clustering (MCL) which are: lack of scalability and the overfragmentation of clusters, authors of Satuluri and Parthasarathy (2009) propose Multilevel Regularized (MLR-MCL). This algorithm (see Algorithm 1) is based on Markov Clustering, which is a flow-based graph-clustering algorithm.

Algorithm 1: Multi-level Regularized MCL Algorithm

Data : Original graph G, Inflation parameter r, Size of coarsest graph c

1 Phase 1: Coarsening
2 Coarsen graph successively down to at most c nodes.
3 Phase 2: Curtailed Regularized-MCL along with refinement.
4 Initialize M to canonical flow matrix of the coarsest graph G_l.
5 Starting with the coarsest graph, iterate through successively refined graphs.
6 Run Regularized-MCL for a small number of iterations.
7 Project flow from the coarse graph G_i onto the refined graph G_{i-1}.
8 Canonical transition matrix of the refined graph G_{i-1}, for the next round of Curtailed Regularized-MCL.
9 Phase 3: Run Regularized-MCL on original graph until convergence.

Result: Interpret M as a clustering.

In the partitioning phase, MLR-MCL operated first for a small number of iterations on the contracted graph. Then the flow from the coarser graph G_i is projected back using MLR-MCL until reaching the original graph which will be performed up to convergence. This enhanced version succeeds in ameliorating the scalability of MCL (it is 2–3 orders of magnitude faster). It also outperforms METIS and Graclus in term of clustering quality. The time complexity of MLR-MCL is $\mathcal{O}(f|\varepsilon| + \sum_{i=1}^{|V_c|} d_i^2)$ (Satuluri and Parthasarathy 2009) with f is a small constant and V_c is nodes of coarsened.

The main difference between multilevel methods cited above (ML, METIS, Graclus and MLR-MCL) is limited simply to the heuristic used when choosing unmarked neighboring nodes. The first approach chose this node in a random way (Random matching) (Hendrickson and Leland 1995). This way is simple, effective and minimizes the number of levels of coarsening in an eager manner. The second heuristic restricts the choice (Karypis and Kumar 1998a). In fact, it selects the node linked with edge having the highest or smallest weight. Heavy clique matching merges the vertex u with a neighbor v unmarked such that the edge density of the "multinode" (u and v) is the biggest (Karypis and Kumar 1998b). In this case, subgraphs which are cliques or nearly cliques will not be cut by the bisection. Therefore, it guarantees a good bisection. All of these methods of contraction have a linear time complexity.

Blondel et al. (2008a) proposed an algorithm named Louvain which has a "Bottom-up" agglomerative aspect. This algorithm alternates between coarsening and partitioning whereas the refinement phase is mostly trivial. Initially, all vertices belong to different partitions, and then, they are contracted in every iteration into a multinode of optimal modularity. Denoted by $Q(C_i)$, the modularity (Newman and Girvan 2004) of a clustering C_i is expressed as

$$Q(C_i) = \frac{1}{d(V)}\left(d_{\text{in}}(C_i) - \frac{d(C_i)}{d(V)}\right) \tag{2}$$

where $d(V)$ is the total degree of the entire graph (i.e., $d(V) = \sum_{v \in V} d(v)$). Each group of parallel edges is aggregated with one weighted link. Once arrived at a first optimal situation, the process passes at the upper level: Every group is treated as a vertex. The process continues until it has no more possible gains of modularity. Then the partitioning phase finishes. The refinement stage consists of releasing the original vertex of multinodes and assigning to them the same community label. The time complexity of Louvain is in $\mathcal{O}(n \log(n))$. This method allows escaping from the defect of the modularity known under the name of "resolution limit". It is the fact that small communities tend to be merged with bigger ones. However, its result is sensitive to the order of the processing of the vertices. Thus, a variant of Louvain (with multilevel refinement) is suggested in Noack and Rotta (2009) which tends to give better results for large graphs.

The study in Waltman and van Eck (2013) used the method of SLM (Smart Local Moving) which detects groups in the wider graphs by maximizing the modularity. This one establishes itself on the algorithm of Louvain with some improvements. In fact, SLM constructed subnetworks

formed by only nodes belonging to a specific community of interest and used the local moving heuristic to identify clusters. The community structure obtained corresponds to the reduced network. Due to this improvement, groups can be disconnected and vertices can move from one community to another. In this way, SLM gives generally higher values of modularity, however, it requires more computing time (Waltman and van Eck 2013).

The most recent method named *Nerstrand* is proposed by LaSalle and Karypis (2015) which is based on the idea of modularity's maximization at all the stages. Indeed, during the first step of the vertices are aggregated to optimize the modularity. This algorithm 2 has explored three different aggregation schemes: MAT (Standard Matching), M2M (Matching with secondary two-hop Matching), FCG (First Choice Grouping). MCCA (Multilevel Coarsening Compact Areas) algorithm is to merge well-connected zones in every level by updating edge and vertex weight until a stopping criterion is met. This study used concept terms, Weighted Node Importance terme to contract compact areas and the re-weighing of edges until reaching a stopping criterion. An extensive experimentation of MCCA demonstrates its effectiveness in both reducing the size of the network and preserving the important graph's properties.

In conclusion, every method is suitable to the law of distribution degree of the network. The initial clustering operates by setting each vertex in a class, then applying agglomerative clustering with the maximization of modularity. In uncoarsening phase, the author uses the clustering of the coarsened graph as an estimate for a good clustering

Algorithm 2: Nerstrand Algorithm

Data : A graph $G_0 = (V, E)$

1 *Coarsening*: A series of increasingly coarser graphs is generated: $G_1, \ldots, G_s$.

2 Initial clustering: A clustering $C = \{G_1, \ldots, G_k\}$ is created by assigning each vertex in Gs to a cluster.

3 Uncoarsening: The clustering C is projected through the series of coarse graphs, $G_s \rightarrow \ldots \rightarrow G_0$, while being improved for each graph.

Result: The clustering C is returned as output.

MAT consists in visiting nodes in a random order and merging them with a neighborhood not yet marked. The condition of merging is the maximization of the variation of modularity. The change in modularity by merging v and u is

$$Q_{\text{merging}}(u, v) = Q_{\{v,u\}} - (Q_{\{v\}} + Q_{\{u\}}) \quad (3)$$

If the vertex has no neighbor, it will be aggregated by itself. This algorithm works well on graphs having an uniform distribution of degree, contrary to the graphs having a power law distributions degrees which prevent big matching. So the reduced graphs have similar size than the initial one.

To solve the problem of MAT, the authors developed a variation of this method (M2M) which uses a secondary jump during matching. M2M has the same principle of MAT except that if all the neighbors of u are marked, it will be merged with one of the neighbors of u. M2M surpasses MAT to generate a reduced graph of small-size (approximately half of the initial size).

Regarding to FCG, if node u is not marked, then its priority for the matching is determined by using the variation of modularity (as MAT). Otherwise, if u belongs to a group of nodes C, then the priority of the addition of v to this group is determined in a similar way (calculate the modularity of the cluster C). This method is adapted to graphs of power law distribution of degree and it is more efficient than M2M.

A novel approach was suggested by Rhouma and Romdhane (2018) for multilevel coarsening. The main strategy of of the graph of high level. Then they improve this partition to find a local maximum modularity. This is repeated until the clustering is implemented and improved to the initial graph. These algorithms run in $\mathcal{O}(m + n)$ time and $\mathcal{O}(m + n)$ space. Nerstand finds clusters of equal or better modularity than existing methods and it is (4.5–27.2) times faster than them (LaSalle and Karypis 2015).

To recap, in this subsection, the community structure exposed by complex real networks often has a hierarchical organization. This suggests that there should be several levels of hierarchy in the graph partition. In other words, there are significant communities at each level. On the other hand, according to a study made by Whang et al. (2012) multilevel framework suffers from several limits. First of all, the success of the algorithms at several levels is based on the hypothesis that the ratio of contraction of the graph is reasonably high at each level. However, for large real social networks, this cannot be the case. More exactly, when a network follows the power of distribution of degree, the rate of the reduction of the graph becomes much smaller. This is due to the innate structure of the network. In fact, when the degree distribution follows a power law, most of the vertices of low degree tend to be attached to vertices of high degree. Consequently, there is a large number of vertices which cannot be merged at every level. Then the graph reduction ratio is small enough on major real social networks. Indeed, a good "coarsening" must hide a large number of edges from one

level to another, which means that the difference between the numbers of links of consecutive levels must be large.

Another important issue is the memory consumption. The multilevel approach generates a series of graphs in the phase of "coarsening", so it requires an additional memory to store the graphs (Whang et al. 2012). When the size of the initial graph is very important, however, it is sometimes impossible to allocate this additional memory. In addition, one of the main drawbacks of this approach is that it depends deeply on the coarsened graph that is why a good coarsening of graph without loss of information is needed.

Finally, the multilevel grouping of large-scale graph presented important obstacles of parallelization (Lumsdaine et al. 2007). First, the "coarsening" is made by contractions of the edge, where every node is authorized to participate in a contraction. To assure this property, a global consensus must be reached between the processes which is harder to realize when the "coarsening" proceeds to a deeper level (the graphs become denser).

3.1.2 Graph reduction

The second category is based on the idea of reduction of the size of the graph in a manageable one, so that the costs of calculation decrease without loss of quality of the solution or the modification of the network structure. The sampling of a graph is a technique of selection of a subset of nodes and/or edges of the original graph (Hu and Lau 2013). It has a wide range of application, for example, the hidden population, surveys in sociology and social graph visualization, etc. In certain scenarios, the whole graph is known and the purpose of the sampling is to obtain a smaller one. In other scenarios, the graph is unknown and the sampling is considered as a way to explore the graph. Techniques usually used are vertex sampling, edge sampling, and traversal-based sampling (2013). We provide here taxonomy of the various existing methods.

Vertex Sampling (VS): This technique selects the first p nodes and retains links between them. The most known methods are: Random node Sampling (RN) (Hu and Lau 2013), Random page Sampling (RP) (Leskovec and Faloutsos 2006) and Random Degree Node (RDN) (Hu and Lau 2013). RN selects nodes uniformly at random, it is the simpler one, but it did not keep the power law distribution of the network and it did not preserve the properties of the original graph. RP is proportional to the weights of the page rank and it gives dense graph. RDN is proportional to the node degree. The number of nodes of high degree is big and the reduced graph is dense.

Edge Sampling (ES): It is the selection of some edges and their end nodes. They include Random Edge (RE) (Hu and Lau 2013), Random node-edge sampling (RNE) (Leskovec and Faloutsos 2006), Hybrid sampling (HYB) (Krishnamurthy et al. 2005). This definition arises only in some theoretical discussion of sampling methods based on graphs. A more realistic definition is to do the sparsification of the graph.

Traversal-Based Sampling (TBS): The sampler starts with a set of initial vertices (and/or edges) then it widens the sample according to current observations. This class of approach occurs naturally in the context of network exploration. Examples are as Snowball Expansion Sampling (2010), Forest Fire (2006), SlashBurn (2011), etc.

The basic idea of Forest Fire (FF) method by Leskovec and Faloutsos (2006) is derived from the action of spreading fire in the woods. In FF, firstly, a node is randomly selected as the center. Then the links and corresponding nodes are burned simultaneously (This manner differentiates FF from others existing methods). If a link is burned, the node at the other end has a chance to burn its own links, and so on recursively. This model has two parameters corresponding to the probability of burning. The sampled graph is obtained when the desired number of nodes is reached or no more nodes can be burned. If the fire stops before having the wished size of graph, FF makes additional runs. The authors assess the quality of samples of these methods according to their ability to match the different properties of the structure of the original graph as: the distribution of degrees, clustering coefficients, and component size. They conclude that the sampled graph obtained by this algorithm is better than other methods (RE, and RN) (2006). In addition, Forest Fire adequate only 15% of sampled graph's size in order to match the original properties of the original graph. The time complexity of FF is no longer linear but is like a tree (Ruan et al. 2015), and this is due to this ability to select and visit multiple nodes at the same time.

The major purpose of Expansion Snowball Sampling (XSN) (Maiya and Berger-Wolf 2010) is to retain samples with good expansion properties which is the most representative of the structure of the original network.

Algorithm 3: XSN Algorithm

Data : A graph $G_0 = (V, E)$, k the sample size

$S = \emptyset$; // initialize sample to empty set

$v = random(V)$; // choose node from V at random

$S = S \cup \{v\}$

while $|S| \leq k$ **do**

  Select new node $v \in N(S)$ based on maximization of:

  $|N(\{v\}) - (N(S) \cup S)|$

  $S = S \cup \{v\}$

end

This concept is derived from the graph theory expansion. XSN is a greedy algorithm (see Algorithm 3) trying to add, at every stage, adjacent (neighbors) vertices which give the maximum expansion factor. We note that the "Expansion Factor" of a set of nodes S having N_S neighbors is:

$$X(S) = \frac{|N_S|}{|S|} \tag{4}$$

New sample members can be selected either in a deterministic or probabilistic (proportional to its improvement of expansion) manner and the process continues until it reaches the size of the desired subgraph. Thus, the sample is developed like a snowball. With a 15 percent of the size of the original network, the authors show that XSN outperforms existing methods for large networks. XNS guarantees that sampled graph contains most groups of initial graph because nodes acting like bridges between clusters form it. On the other hand, this method has high complexity due to its combinatorial nature (2015).

Markov Chain Monte Carlo (MCMC) follows the same principle as previously and every sampled graph is considered as a state in the Markov chain. Contrary to XSN, MCMC aims to obtain the maximal possible Normalized Expansion Factor ($\frac{|N_S|}{V \setminus |S|}$), which achieves every vertex in graph with one hop (Hübler et al. 2008). This algorithm begins by opting for a state formed by random n nodes. At the next "candidate *ith* stage" S^i, MCMC replace in random manner a node in S^{i-1} with a node not belonging to $(V \setminus S^{i-1})$. According to the quality scores, the method decided to pass to the next stage with same substitution. While the stationary state distribution of a Markov chain is attained, the final set S correspond to the sampled graph (2008). MCMC like XSN suffers from a high complexity. Backtracking is one of the disadvantages of this algorithm.

A new approach was proposed to provide an ordering of nodes in the graph called SlashBurn (Kang and Faloutsos 2011). Algorithm 4 shows the high-level idea of SlashBurn.

Algorithm 4: SlashBurn Algorithm

Data : Edge set E of a graph $G = (V, E)$, a constant k (default = 1).

1 Remove k-hubset from G to make the new graph G'. Add the removed k-hubset to the front of Γ.

2 Find connected components in G. Add nodes in non-giant connected components to the back of Γ, in the decreasing order of sizes of connected components they belong to.

3 Set G to be the giant connected component (GCC) of G. Go to step 1 and continue, until the number of nodes in the GCC is smaller than k.

Result: Array Γ containing the ordering $V \rightarrow [n]$.

This method is analogous to burn the hubs, and slash the remaining graph into smaller connected components. It has been used to obtain good compression of the graphs of the real world. It can also be used for obtaining a subset of nodes that contain information about the community's inherent structure. After selecting the k-hub (setting k to 0.5% of the number of nodes decrease the number of iterations), the set of connections is burned and a new graph is built. The giant connected component is discovered in

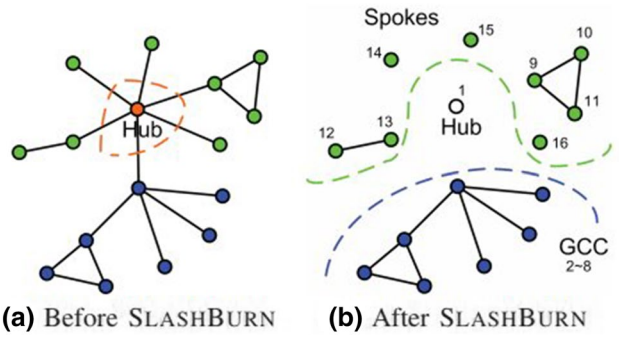

Fig. 5 A graph before and after 1 iteration of SlashBurn

this new graph and the selection process is performed recursively until they reach the required size of the subset. The idea behind SlashBurn is the intuition that in the real-world graph is easily disconnected by hubs. Furthermore, having a small number of denser blocks gives better opportunity for compression. Figure 5 shows a graph before and after 1 iteration of SlashBurn.

Experiments show that this method surpasses others methods in term of quality of compression, running time and the quality of cut. SlashBurn takes $\mathcal{O}(m + n\log(n))i$ time where i is the number of iterations and its space complexity is $\mathcal{O}(V)$ (Kang and Faloutsos 2011).

The big challenge of the graph sampling method is not only having an inexpensive analytical solution of the social network, but how to provide a "realistic" mimic copy of the original graph. In fact, the deeper question is: how do we measure success of sampling method? It appears that there is not a single perfect answer. Depending on the particularity of applying, a suitable algorithm is chosen. Leskovec and Faloutsos (2006) and Wang et al. (2011) point out that maintaining the properties and community structure of the original graph are an index of the success of the graph sampling method. They check a list of properties of the sampled graph (degree distribution, clustering coefficient distribution, the distribution of sizes of weakly connected components (WCC), etc.) and compare it with the value of the initial one. We notice that our objective is to produce sampled network keeping the community structure of the graph.

For this reason, since the sampled graph is obtained from any above-mentioned methods, we can run any community detection algorithms due to the much smaller scale.

The authors of Maiya and Berger-Wolf (2010) compare the quality of partition identified by expansion-based sampling methods (XSN and MCMC). The experiments prove that these methods coupled with the algorithm of detection [Girvan-Newman (GN), eigenvector method (NLE) and greedy optimization algorithm (CNM)] produce a good partition with limited run time.

In the work in Satuluri et al. (2011), the author discusses how to sparsify a graph, that is to say, how to reduce the edges while keeping intact nodes to allow faster classification of vertices without loss of quality. The main idea behind this approach (*LSpar*) is to preferentially retain the edges that are likely to be part of the same group. It proposes to classify the edges using heuristics "similarity-based sparsification" (technique Minwise). Indeed, edges connecting nodes sharing many neighbors are retained. The time complexity of this method is $\mathcal{O}(mk)$ where k is the number of minwise of all the nodes in the graph and m is the number of edges in the graph. To evaluate this method, authors compare the performance of *LSpar* with RE and FF. This pretreatment phase is accomplished by running the induced graph in for community detection algorithms. Results show that using *LSpar* speeds up graph-clustering algorithms without sacrificing quality.

The work of Sadi et al. (2010) (Ant Colony System (ACS)) is an improvement of its algorithm quoted in 2009 through following various processes. First of all, the authors apply the sampling "Snow Ball" to create subgraphs of the original graph according to the predetermined amount of threads. Further, the model ACO is used to select the quasi-cliques (almost a clique) subgraphs in a parallel way. The resulting overlapping cliques will be clusters of the next step. All the subgraphs are transformed into a new reduced graph (with links created based on the concept of "betweenness"). An algorithm of detection of communities is applied to the transformed graph to find communities. The experimental results on the various static social networks with average sizes proved that the global quality of the solution is preserved and the rate of reduction of the graph is important. Contrarily, this method has a high complexity due to the search of the cliques; it requires previous knowledge of some parameters that are not obvious.

The main object of graph extraction of weighted kernel k-Means "GEM" (Whang et al. 2012) is the extraction of a skeleton of the initial graph. This reduced shape consists of nodes of high degree (which represents the most popular and influential people in networks). The next step is to group the vertices using the algorithm of "kernel K-means". Finally, the process ends with a result of the propagation phase of the entire network and the refinement phase. The major inconvenience of "GEM" is that it is parametrized. So, the phase of the extraction of the skeleton of the graph is not proved in theory.

To reduce the huge size of the graph and preserve its properties at the same time, many methods are proposed. These approaches, while being very rigorous, it is NP-hard. In addition, there is no theoretical proof that the real source network structure is preserved. Another disadvantage of these methods is that they generally require knowledge of the global graph, which is impossible in certain scenarios

(in decentralized social networks). Besides, these methods are incapable of facing the dynamic nature of the network and exploiting from zero the network of every snapshot. At the end, these methods depend on the law of distribution of the degree of the network. In conclusion, the reduction of the size of a graph is a simple and intuitive approach. This is typically one of the first thoughts to process massive data. It was widely used, but the performance is not systematically studied. Thus, the theoretical analysis of its methods can establish a more solid basis for future works.

3.2 Centralized community detection based on structural and semantic partition

While many community detection methods have been proposed, which primarily explore network topologies, they provide little semantic information of the communities detected. By integrating the semantic information, a great potential for community detection is held. Here, each approach is described separately.

The early work by Neville et al. (2003) could be considered as the first work studying the community detection method for graphs with attributes. They use a measuring similarity to modify the weights of the edges, then they detect communities through using MonteCarlo or k-means approach. The number of attributes in common determines the similarity ($S_{i,j}$) between two nodes, which are conditioned by the existence of edge between them. The similarity measure $S_{i,j}$ of two objects i, j and k attributes is defined by:

$$S_{i,j} = \begin{cases} \sum_k S_k(i,j), & \text{if } e_{i,j} \in E \text{ or } e_{j,i} \in E \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

Where

$$S_k(i,j) = \begin{cases} 1, & \text{if } k_i = k_j \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Once the weights are modified and standardized, each node is assigned to a community. The process starts with considering each node a community. Thereafter, edges are selected randomly according to their weights and the corresponding nodes are merged into a community. This process is repeated until a stopping criterion is met.

A similar approach is presented in Steinhaeuser and Chawla (2008). This method uses the Jaccard coefficient as similarity measure between attributes to define the weights of the edges. The communities merge process is performed by taking each of the edges and by evaluating whether the weight is above a threshold. Finally, the quality of the final partition measured using the proposed modularity (Newman and Girvan 2004).

Ge et al. (2008) presented the method Connected k Centers (CKC) to find k communities in a social network by imposing constraints on two groups: a constraint on node semantic similarity to the manner of *k-means*, along with a connectivity constraint. The attributes of the actors are defined as a function $w : V \rightarrow R^D$ that is to say, that each of the nodes is characterized by a feature vector of dimension D. The method begins by randomly selecting k nodes to be used as cluster centers. Then each of the other nodes is assigned to a group; for this assignment, the graph is traversed in width thereby maintaining connectivity constraint, keeping the similarity between the players with a measurement for the attribute vectors. These steps are repeated until we get stable groups.

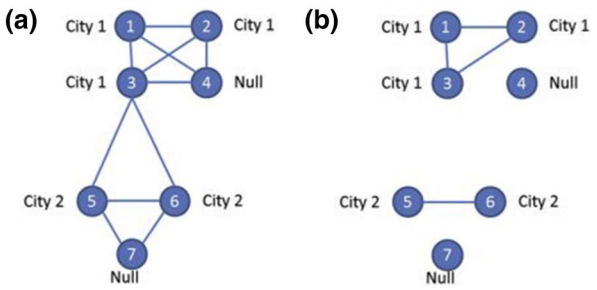

Fig. 6 An example of additional network construction. **a** The friendship network; **b** the network of current city

Zhou et al. (2009) proposed an approach that combines structural and attribute similarities with a unified distance measure. To connect nodes, which share attributes values, attribute nodes and edges are added to the original graph. A neighborhood random walk model is used. A clustering algorithm SA-Cluster is proposed to measure the node closeness on the modified graph, following the K-Medoids method. The time complexity of the algorithm is $\mathcal{O}(n^3)$.

Dang and Viennet (2012) offer two community detection approaches. The first approach is to modify Newman's modularity function. In the beginning they partition the graph into k groups, then apply the modularity of Newman's. This method is chosen because of its scalability. Then, they set a new function, modularity attribute between nodes and they finally introduce the modularity composite as a weighted combination of the modularity structure and modularity attributes. The second step is to find an approximate optimization with a modularity approach inspired by the Louvain algorithm, which considers that each node is a community. A node is then chosen at random. The algorithm tries to move the node from its current community. If a positive gain is found, the node is then placed in the community with the maximum gain. Otherwise, it remains in its original community. This step is applied repeatedly until no improvement is obtained. The first approach, repeatedly checks all nodes, leading to $\mathcal{O}(n^2)$ complexity. To reduce the computational cost, they propose another approach that only makes use of node's nearest neighbors. Considering an attributed graph,

the first step is to build the k-nearest neighbor graph as a directed graph where each node has exactly *k-edges* connected to its most similar neighbors. In the second stage, various methods can be used to find communities. In this approach they chose Louvain method for detection, because of its scalability. This approach uses $\mathcal{O}(n^2)$ time and $\mathcal{O}(n \times k)$ space.

Wang et al. (2014b) propose a method that allows discovering and profiling of overlapping communities based on location social networks (LBSNs). They take into account the overlapping appearance: They consider that a user can have multiple preferences and may be interested in different communities. For each network, they consider two types of nodes: users and locations, and two check-in network represents the edges between these two types of nodes. For community detection, they consider not only the check-in network (characteristics inter-nodes) but also the attributes of users and locations (characteristic intranodes). The first step is to collect selected features from a dataset (LBSNs): perform normalization and merging of these characteristics. The second step consists in detecting the structure of overlapping communities through one of the proposed clustering algorithms. Finally, a combination of community detection and meta-data user's locations gets community profiles that interpret the social and semantic meanings of communities. In this way, each community shows explicitly "that is" interested in "where", with which "attributes", which is very useful for real applications.

Another method is proposed by Fan and Yeung (2014). The authors introduced community detection methods for online social network incorporating the profile information's in Facebook networks as an example. It will be shown that profile information of users is useful in community detection. Each user has a profile to record their basic information. For example, when people create accounts on Facebook, they are required to provide their names, email addresses, birthdays, and other information. If they like, users may also provide additional information (e.g., their locations, colleges and so on). All these pieces of information are shown on their profiles, so, we can obtain some features of these users. As members of the same community may share some common features, it is therefore possible that information on profiles is helpful for community detection. In this proposed method, the profile information is used to construct additional networks, where all nodes are kept, and a connection between two nodes will be added if the two following conditions are met: Both nodes have a connection in the network of friendship and the "features taken into account" of these two nodes are identical (see Fig.6).

Finally, they run the algorithm for maximizing modularity, (extensions on Newman's modularity matrix method or Louvain method) with additional networks.

A novel semantic community identification method, SCI, to detect network community structures and infer their semantics simultaneously is proposed by Wang et al. (2016b). This method based on a novel nonnegative matrix factorization (NMF) model with two sets of parameters; the community membership matrix and community attribute matrix, and present efficient updating rules to evaluate the parameters with a convergence guarantee. SCI is formulated as an optimization problem in a novel nonnegative matrix factorization (NMF). NMF consists of two sets of parameters: the community membership matrix and community attribute matrix, and presents efficient updating rules to evaluate the parameters with a convergence guarantee. SCI is its ability to semantically or functionally annotate each of the identified communities.

Recently, Chaabani and Akaichi (2017) addressed the community detection problem by a novel framework called *Mean*CD based on function measuring the strength of links describing the information (topics) shared between users. The function is given as

$$J(T_{V_i}, T_{V_j}) = \frac{T_{V_i} \cap T_{V_j}}{T_{V_i} \cup T_{V_j}} \tag{7}$$

where T_{V_i} represents the subjects of User's interest V_i and T_{V_j} denotes the user's topics V_j.

More recently, a multiobjective evolutionary algorithm based on structural and attribute similarities (MOEA-SA) is first proposed to solve the attributed graph-clustering problems by Li et al. (2018). Two objectives termed as modularity Q (see Eq. 2) and attribute similarity S_A are used to be maximized in the algorithm. S_A is defined as a criterion to measure the quality of attribute similarity of nodes inside clusters of G. S_A is given as

$$S_A = \frac{\sum_{k=1}^{c} \sum_{i,j \in k, i<j} 2s(i,j)}{\sum_{k=1}^{c} r_k(r_k - 1)} \tag{8}$$

where c is the total number of communities, $s(i, j)$ is the similarity of nodes i and j defined as $\frac{\vec{i}.\vec{j}}{|\vec{i}| \times |\vec{j}|}$ and r_k is the number of nodes inside community k. MOEA-SA uses a hybrid representation to make full use of the relationships between vertices. Moreover, an efficient operator named as multi-individual-based mutation is designed, and accompanied by a neighborhood correction strategy. The hill-climbing strategy is used to overcome the slow convergence of modularity Q. The experimental results demonstrate the effectiveness of MOEA-SA and the systematic comparisons with existing methods show that MOEA-SA can get better values of density and entropy in each graph and find more relevant communities with practical meanings.

4 Centralized community detection in dynamic social networks

4.1 Centralized community detection based on structural partition

Because of the major challenges caused by the enormous size of the graph and its dynamic evolution over time, only a limited number of methods have been presented to deal with this problem. These algorithms are categorized into two families: (1) approach applies a standard static methods directly in every snapshot and it matches the founded communities with those of the previous snapshot. (2) looking for communities by studying the whole network, the incremental snapshots approaches and methods directly studying the temporal network.

Hopcroft et al. (2004) suggest a method of hierarchical agglomerative detection that is interested in the dynamic appearance of communities (contraction and expansion). They calculate the pairing between the two communities in two instances. This matching is defined as:

$$\text{match}(C_1, C_2) = \min\left(\frac{|C_1 \cap C_2|}{|C_1|}, \frac{|C_1 \cap C_2|}{|C_2|}\right) \quad (9)$$

The instability of the community detection algorithm (many communities that largely disappear or change between two times) makes the monitoring phase of communities impossible, since the changes are due to the algorithm and no more to a modification of the network structure. To limit the problem of instability of the detection, the authors consider only the changes of natural communities (communities which even if we introduce a minor change of the network, remain identical). This restriction eliminates many interesting communities. Also, the extraction of significant communities of the hierarchical tree is not a trivial task.

Greene et al. (2010) propose a scalable model to detect and track the community structure of the dynamic network composed of multiple time steps. Each cluster is characterized by a sequence of significant events. The community's detection algorithm used in every snapshot is MOSES (McDaid and Hurley 2010), because it is able to manage overlapped clusters. Next, a matching system is used to recognize communities of different snapshots. This method uncovers networks of big size with variant evolutionary characteristics.

These methods are the simplest ones, since they involve only static algorithm of community detection, followed by a post-treatment. However, they suffer from the instability of clusters. Also, they neglect historic information, which causes a redundant computation and high complexity in megascale social networks.

Basing on the history of the network and its evolution, Quick Community Adaptation (QCA) (Nguyen et al. 2014) is able to discover and track the community structure of a dynamic network. This modularity-based framework uses the method proposed by Blondel et al. (2008a) to detect the initial disjoint static partition of the network (fast algorithm). Then it updates clusters when a significant change occurs. These changes can be inserted or removed one (or a set) edge (or node). By adopting three fold intuitive scenarios QCA proposes a theoretically based algorithm suitable for every change. In fact, adding edges intra-clusters or removing inter-clusters links consolidate the structure of the community. Vice versa, this structure is lost. Also, if two communities are attractive to each other by adding interactions, there is a possibility that they will be combined to form a new community. QCA demonstrates its effectiveness in both synthesized and real networks in regards to the modularity scores and the quality of partition in a timely manner (Nguyen et al. 2014). Yet, this method is computed in every simple event. In fact, authors process network changes as a set of simple events which causes low efficiency (He and Chen 2015).

Another attempt that is closely related to QCA method is presented in Facetnet (Lin et al. 2009). It is a solution based on a probabilistic generative model. This method consists in formulating a function quality similar to a factorization problem of nonnegative matrices which jointly optimizes the quality and stability of communities. Although this method has the advantage of enabling the detection of overlapping communities, it nevertheless imposes strong constraints: the number of communities should be known in advance. Also, it is not possible, in a priory, to add or remove nodes over time. It does not allow either of operation such as merging or dividing communities. Another limit of Facetnet is that at each time step, this algorithm must be executed for several values of the number of communities, which could prevent it to be effective on real social networks.

To reduce computational cost, the method presented by He and Chen (2015) considers only the changes of the network. Indeed, where interactions between members of the same community are kept in $(t-1)$ and t they still in the same cluster at t. The steps of this method are: Firstly, at every snapshot a small network is constructed by considering communities of previous time step. If the vertices of a group don't change over time they are treated as a meta-node with a self-loop. In the contrary state (node's connections swap), this node is considered as a single node in the new graph. Secondly, the authors apply the fast greedy algorithm of Blondel et al. (2008a), based on the modularity optimization, to detect clusters in the sampled network. Experiments showed that this method improves 69% of CPU running time with the maintenance of the quality of partition. Its time complexity is $\mathcal{O}(\alpha * m)$ where α is the number of snapshots

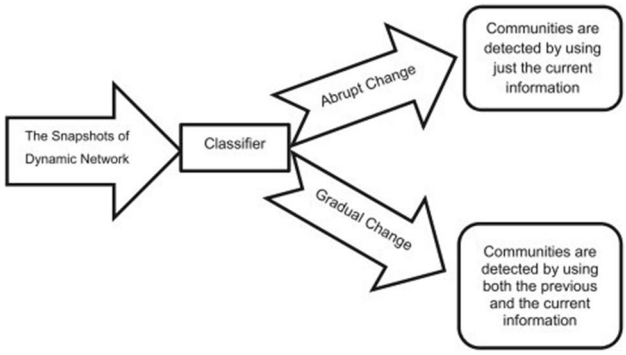

Fig. 7 The change-aware model scheme

(He and Chen 2015). The main drawback of He and Chen's method is that they consider only the change of the edge of the network.

The dynamic smart local moving algorithm (DSLM) proposed by Aktunc et al. (2015) is a modified incremental dynamic version of SLM (Waltman and van Eck 2013). In the beginning, it uses the historical results of SLM to assign nodes to the initial communities. After some iteration, the value of modularity is down. This causes a stop of the algorithm using either the amount of changes or setting a threshold value of modularity. This led to decreasing the running time of DSLM (see Algorithm 5). Also, it can decrease the number of iterations to converge.

Algorithm 5: dynamic SLM Algorithm

```
Data : Old Communities, Old Network, New Network
for i = 0 → Old Communities.size do
    New Communities = ReadCommunitiesFile()
end
Delta Network = New Network - Old Network
for j = i → DeltaNetwork.size do
    new communities[j]=Delta Network[j-1]
end
Result: New Communities.
```

Bhat and Abulaish (2015) developed a new algorithm HotTracker which discovers a preliminary community structure of an initial state of the network using a density-based algorithm (a group is discovered by detecting the neighborhood of each object in the graph). Then for every new state of the network it uses only active nodes to manage the dynamic aspect. The authors apply an approach based on logging intermediate transition events to map the evolution between two successive snapshots. This reduces the comparisons between communities. HotTracker is applied in different types of graphs: directed, undirected, weighted and simple and detects overlapping communities. Its time complexity is $\mathcal{O}(n^2)$.

The first approach that tried to apply machine learning in the dynamic detection of communities is learning-based targeted revision (LBTR) (Shang et al. 2016). The primordial step of this framework is the construction of the initial partition of the network (state t_0) applying the local modularity algorithm Louvain (Blondel et al. 2008a). Once the network evolved from t_0 to t_1, LBTR changes the assignments of nodes and their neighbors to maximize the gain of modularity. However, different from the existing methods, it does not check every new node. For this, the authors use a Learning-based targeted revision to decide whether the new movement needs a revision of the community structure or not. To tackle the problem of over subdivision of the graph, this method tries to merge small clusters (size less than a threshold) with their neighbors communities. LBTR can reduce the running time with conserving of the quality partition. Furthermore, the performance of this method still stable over time when it uses real and artificial benchmarks. Its time complexity is $\mathcal{O}(|\Delta V|)$, where $|\Delta V|$ is a number of new and changed vertices from $(t-1)$ to t.

Recently, Samie and Hamzeh (2018) proposed a new approach by detecting abrupt changes as an event in social networks by using the least information of the network and employing some new features which are very simple and efficient. The new framework for community detect the change type gradual and abrupt, and then decided how to discover the communities of the current snapshot, with or without considering the information of the previous one(s). The change-aware model scheme is shown in Fig. 7

In conclusion, the clustering is one of the essential tasks for the exploration of data and it needs to improve nowadays more than ever, to help the analysts to extract knowledge of massive and dynamic data. Several approaches, particularly the older, had considered dynamic graph as an independent succession of snapshot. These methods have an advantage: they allow reusing the unchanged existing static methods, but they suffer from instability problems (between two consecutive snapshots, communities will be different). Other algorithms make use of modularity. They are looking to optimize each snapshot of the dynamic graph or to maintain it at a high level over the changes. But Fortunato (2010) proved the resolution limit of the quality measurement (over subdivision of the network). Another negative point is that while the overlap is an important phenomenon, several previously cited methods do not take it into consideration.

4.2 Centralized community detection based on structural and semantic partition

Recently, some of the researches were interested in community detection in a dynamic social network by taking into

account the structure and content of edges and nodes. In fact, this case is more real and logical because the network is not just solid or static graph, but there is a huge evolution of its form and content. This content represents the user interests and thought, some published or forwarded message between users. This domain is in its first steps.

Authors of Osborne et al. (2014) presented an algorithm called Research Communities Map Builder (RCMB) able to discover topic-based communities over time (people working on semantically related topics). To begin, authors sum semantic topic vector of publications of researcher in each year according to the formula:

$$\mathrm{CT}(T) = \sum_{i=1}^{n} P(T|ct(i,T))^{\varphi} \tag{10}$$

where $ct(i, T)$ indicates the set of topics associated with the ith publication that is in a *contributesTo* relationship with T. $P(T|ct(i, T))$ is the probability that a paper with such a set of topics is also explicitly associated with area T at the publication date of the ith paper. Then, they detect fuzzy communities using FCM algorithm (Bezdek et al. 1984) on the weighted semantic generated vectors. At the end, it computes the similarity between each couple of communities to predict their interaction over time. RCBM allows a finer understanding of the evolution of clusters on time (the appearance and fading). This work is a new solution in semantic technologies and data mining to help users to explore and make sense of science data.

NEIWalk (Algorithm 6) is one of the recent method (Wang et al. 2014a) which detects overlapped dynamic communities.

Algorithm 6: NEIWalk Algorithm

1 Map the original content-based network into NEI network.
2 Perform the initial clustering based on hitting time.
3 Capture the evolving nodes and edges for the new network.
4 Update the NEI network.
5 Perform heterogeneous random walk to update community labels.

The original content-based network is transformed to Node-Edge Interaction (NEI) network using "similarity metric" (a multimode network comprising two types of nodes and three types of edges). NEI kept linkage structure, node and edge content. At the initial instance, the static NEI is categorized using the similarity measure calculated from hitting time between pairs of n-nodes. The node content similarity between two n-nodes v_i and v_i is given as:

$$\mathrm{sim}_v(v_i, v_j)(S) = \frac{f_v(v_i)^T . f_v(v_j)}{\sqrt{f_v(v_i)^T . f_v(v_i)} . \sqrt{f_v(v_j)^T . f_v(v_j)}} \tag{11}$$

where $f_v(v_i)$ is the d-dimensional feature vector used to represent the node content. In the following instance, heterogeneous random walk is applied to compute dynamic communities in the updated NEI. Experiments in two real-world dynamic social networks have shown the effectiveness and efficiency of this algorithm.

Unprecedented growth in social networks creates a significant quantity of data. Tag assignments stream clustering (TASC) (Wu and Zou 2014) try to distill temporal communities of such data where tags and interactions between user are included. To handle this, users profile vectors were generated in order to represent user's interests. To discover communities of users, TASC used LSH-based (Charikar 2002) method which identifies similarities among users. The next challenge is to update these clusters over time. For this the similarity between the new user and his nearest neighbor is recalculated at every modification. If its value is greater than a threshold, a user is added to the cluster. Otherwise, a new community is created. Contrary to other methods, the time complexity of TASC linearly increased because the time of processing one tag assignment was constant.

A novel method called temporal overlapping community detection using Time-weighted Association Rules (TOTAR) is proposed in Feng et al. (2015). The leading idea of this method is incorporating temporal factor in the graph partition and associating used rules. The three main stages of TOTAR are: Firstly, it applies a local fitness maximization algorithm having an overlapping aspect (LFM) (Lancichinetti et al. 2009) to detect communities from a user–user interaction graph. The fitness function is given as

$$f(c) = \frac{w_{\mathrm{in}}^c}{w_{\mathrm{in}}^c + w_{\mathrm{out}}^c} \tag{12}$$

where w_{in}^c and w_{out}^c are the internal and external weights of community c. Then, it merges clusters having too many common nodes. The links of this graph represent the common preferred items between two users and its weight corresponds to the number of shared items. To incorporate the effect of interest changes, the authors adopt the idea: the more closely transactions of two users have over time, the more coincidence in interest they will have. Designing a time-weighted association rule mining algorithm to shape the drift of user interest is the second step of this framework. At the end, TOTAR proposes a dynamic overlapping recommendation model. This framework proves its efficiency compared to existing methods, but it has some drawbacks. In fact, the choice of starting vertices in LFM algorithm affects the partition of the network. The consideration of similar dislikes recommendations involves an improvement in the model's result.

In conclusion, some recent efforts have been made to integrate structure and topic when discovering communities in

Table 2 Summary table of centralized community detection

Method	Detection nature	Network nature	Predefined community number	Overlap	Complexity	Advantages	Drawbacks
ML (1995)	Structural	Static	Yes	No	–	Performs spectral bisection Coarsened graph approximates initial one Powerful in continuous mathematics	The construction of a sequences of graphs is memory intensive RM is quite unpredictable
METIS (1998a)	Structural	Static	Yes	No	$\mathcal{O}(m)$	Linear complexity time	Produce communities of equal size Sensitivity to parameters
Graclus (2007)	Structural	Static	Yes	No	$\mathcal{O}(kn)$ k: number of clusters	The refinement step converge very fast An eigenvector-free method Do not produce communities of equal size	Sensitivity to parameters
MLR-MCL (2009)	Structural	Static	Yes	No	$\mathcal{O}(f\lvert\varepsilon\rvert + \sum_{i=1}^{\lvert V_c\rvert} d_i^2)$ f: a small constant V_c: set of nodes of coarsened	Save memory and computation Do not produce communities of equal size	Sensitivity to parameters
Louvain (2008a)	Structural	Static	No	No	$\mathcal{O}(n\log(n))$	No resolution limit Fast	The order of node selection affects the computation time
SLM (2013)	Structural	Static	No	No	–	Improve the modularity quality	More computing time than Louvain
Nerstand (2015)	Structural	Static	No	No	$\mathcal{O}(m+n)$ temporal complexity $\mathcal{O}(m+n)$ space complexity	High quality of partition 4.5–27.2 times faster than existing methods	Resolution limit of modularity
ACS (2010)	Structural	Static	Yes	Yes	–	Important rate of reduction	High complexity (search of the cliques) Needs previous knowledge of some parameters
GEM (2012)	Structural	Static	Yes	No	–	Reduced run time	The skeleton is not proved in theory
Wang et al. (2014b)	Structural and semantic	Static	Yes	Yes	$\mathcal{O}(n^2)$	Multimode and multiattribute edge-centric co-clustering framework	Overlapping communities
Fan and Yeung (2014)	Structural and semantic	Static	No	No	With Louvain method is $\mathcal{O}(n)$ With Newman method is $\mathcal{O}(n^2 logn)$	Highest modularity Partitions more meaningful Better quality of detection	Noises in additional networks Some missing of useful information
Neville et al. (2003)	Structural and semantic	Static	No	No	–	Robust to irrelevant attributes	Less robust to irrelevant links
Chaabani and Akaichi (2017)	Structural and semantic	Static	No	No	–	Simple method Meaningful results	Overfragmentation of the network

Table 2 (continued)

Method	Detection nature	Network nature	Predefined community number	Overlap	Complexity	Advantages	Drawbacks
Zhou et al. (2009)	Structural and semantic	Static	No	No	$\mathcal{O}(n^3)$	Very good balance between structural and attribute similarities Partitions a graph with considering connectivity	Downgrade the intra-cluster cohesiveness
Dang and Viennet (2012)	Structural and semantic	Static	No	No	$\mathcal{O}(n^2)$	Provide meaningful communities Flexibility in combining structural and attribute similarities	Higher complexity
Ge et al. (2008)	Structural and semantic	Static	Yes	Yes	$\mathcal{O}(n^{k+2})$ k: number of random centers	Meaningfulness results Efficient heuristic algorithm	High complexity
Hopcroft et al. (2004)	Structural	Dynamic	Yes	No	Depend on the used Com. Det. algo.	Sample	Eliminates many interesting communities High complexity (sequences of instances)
Greene et al. (2010)	Structural	Dynamic	No	Yes	–	Sample	Instability of clusters High complexity (sequences of instances)
He and Chen (2015)	Structural	Dynamic	No	No	$\mathcal{O}(\alpha * m)$ α the number of snapshots	High quality of partition Reduce computational cost	Consider only change in link
Facetnet (2009)	Structural	Dynamic	Yes	Yes	–	Sample	The number of communities should be known in advance No merging or dividing communities It is not possible to predict the future behavior of the individuals of network
QCA (2014)	Structural	Dynamic	No	No	–	Timely method	Computed in every sample event low efficiency
DSLM (2015)	Structural	Dynamic	No	No	–	Good quality of partition	Modularity resolution limit
HotTracker (2015)	Structural	Dynamic	No	Yes	$\mathcal{O}(n^2)$	Good performance (time and quality)	Noise sensitivity
LBTR (2016)	Structural	Dynamic	No	No	$\mathcal{O}(\lvert\Delta V\rvert)$ $\lvert\Delta V\rvert$: number of new and changed vertices from $(t-1)$ to t	Good performance(time and quality)	Noise sensitivity
NEIWalk (2014a)	Structural and semantic	Dynamic	No	Yes	–	Effectiveness and efficiency	A bounded accuracy loss due to the random walk sampling

Table 2 (continued)

Method	Detection nature	Network nature	Predefined community number	Overlap	Complexity	Advantages	Drawbacks
RCBM (2014)	Structural and semantic	Dynamic	Yes	Yes	–	Explore and make sense of science data	Sensitivity to parameters
TASC (2014)	Structural and semantic	Dynamic	Yes	No	–	The time complexity increases linearly Efficient and effective	Sensitivity to parameters
TOTAR (2015)	Structural and semantic	Dynamic	Yes	Yes	–	Good performance (time and quality)	Seeding nodes affects the partition of the graph

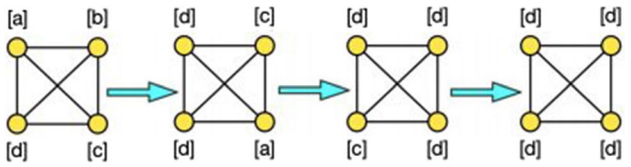

Fig. 8 Illustration of the label propagation with synchronous updating

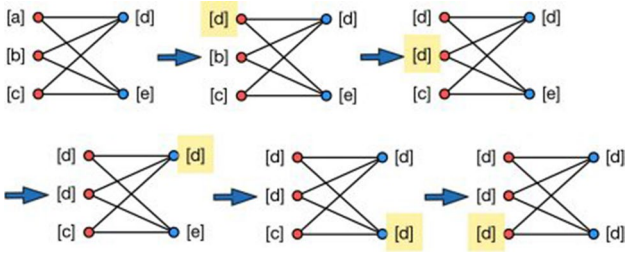

Fig. 9 Illustration of the label propagation with asynchronous updating

dynamic social networks. Although detected communities are too meaningful, utilizing this technique (coupling node/edge content and structure linkage) reside a big challenge.

The first part of this survey which contains the different centralized communities detection methods is culminated by an overview Table 2.

5 Distributed community detection in static social networks

As observed above, several centralized community detection is a well-studied problem in the field of network science. These methods detect communities in a sequential manner. According to a given strategy, such as optimization methods, divisive clustering algorithms, an objective optimization problem and a quantitative function, these methods detect one community at each iteration. Stated otherwise, the community mining process is always serial. However, this centralized and global control is a significant bottleneck when used in very large-scale networks because of high time complexity or no a priori knowledge. In the literature, many efforts have been taken to find various algorithms to efficiently and effectively address the community mining in large-scale networks. Recently, instead of centralized mining network communities, several new community mining algorithms based on distributed detection are proposed. The task of distributed community mining can be distinguished into two different categories. The first and most studied adopts a distributed detection framework with shared-memory parallelism. Several approaches propose a flexible and extensible community detection framework with shared-memory

parallelism (Zhang et al. 2009; Riedy et al. 2011; Gregori et al. 2013; Kuzmin et al. 2013; Staudt and Meyerhenke 2013; Saltz et al. 2015). On the other hand, the second category—which is the focus of this survey—refers to the task where the goal is to apply a range of algorithmic techniques and intuitions to extract several communities in each iteration in an undistributed system instead of extracting the community one after the other. Algorithms for distributed community detection, have garnered significant attention in the past decade. In the literature, few different approaches have been proposed to find community structure in networks in a distributed manner. However, there is no state-of-the-art survey investigating the strategy and performances of techniques for solving distributed community detection in large-scale social networks. There is no guidance to help researchers in comparing or selecting distributed detection techniques. Hence, a guidance and comparative study is necessary. This research analyzes the methodologies significance in addition to the evaluation and comparison of alternative detection approaches performance. Analysis is based on key aspects concerning the evaluation and comparison of alternative distributed detection approaches performance, that is, the efficiency for volume, complexity, and especially the efficiency to overcome the problems of distributed detection, such as coordination and coherence.

The current analysis part is performed on the distributed detection in static social networks: based on structural (in Sect. 5.2) and based on semantic (in Sect. 5.2) partition.

5.1 Distributed community detection based on structural partition

This analysis part focuses on approaches that are based on the structural side network in *distributed* detection. Following the method used, the structural-based distributed detection are reviewed and categorized into three categories which reflects how communities are identified. These categories are described as follows: (1) Label propagation algorithm, (2) Self-aggregation and self-organization algorithm and (3) A multiagent-based decentralized algorithm. Following this classification, each of these categories is presented subsequently.

5.1.1 Label propagation algorithm

The label propagation algorithm (LPA) or "epidemic" community detection algorithm, was first applied in the community detection field by Raghavan et al. (2007). This algorithm, which has become one of the most widely used algorithms in community detection, is practical and simple. The biggest advantage of LPA is its near linear time complexity $\mathcal{O}(km)$, where k is the number of iterations. The algorithms LPA and these variants can naturally be viewed as distributed algorithms, meaning each node only has local knowledge, i.e., knowledge of its label and the labels of its neighbors obtained by means of message passing along edges of the networks. Distributed algorithms are generally classified as synchronous or asynchronous algorithms. The main idea of the algorithm can be summarized as the diffusion of information to identify communities (shown in Figs. 8 and 9). At the beginning of the first phase, each vertex is initialized with a unique label. Subsequently, the algorithm progresses from one iteration to another and at every iteration each node updates its current label according to a given protocol. An iterative process of LPA converges when no nodes change anymore. Finally, groups of nodes sharing the same label are classified into the same community. An iteration ends once all vertices are linearly scanned in this fashion. Consequently, LPA updates multiple nodes at the same time to finally have several communities at the same time.

Starting by the algorithm of Raghavan et al. (2007), a node updates its label with a label shared by the maximum number of its neighbors. Within each iteration, the algorithm linearly scans the vertices label and determines the dominant label among those neighbors for each vertex. Once the dominant label is determined, the vertex label updates the current label with the dominant label. Thus, densely connected sets of nodes form a consensus on some particular label after few iterations. Unfortunately, a convergence problem is presented. For example, consider a bipartite network with two sets of nodes, denoted first set with label "a" and second set with label "b". At some point of the algorithm, the first set will share label "b" and all second set share label "a". Moreover, in next iteration, all nodes will recover their initial labels and the algorithm fails to converge. In order to solve the label oscillations to ensure convergence, Raghavan et al. (2007) have proposed the asynchronous update of nodes in random order that severely hampers its robustness and the stability of the identified community structure. The stability of LPA has become a severe issue and further improvement has been done in various ways (Šubelj and Bajec 2011). With LPA, 95% of nodes or more are classified correctly in the end of iteration 5 (2007). LPA is an efficient linear algorithm. Due to its simple local and decentralized process, LPA is applicable to large-scale networks with millions of nodes and edges. The advantage of a LPA is, in addition to its simplicity, the fact that it can be easily parallelized or distributed. Contrary, some drawbacks of LPA is the low algorithm stability. This low stability is due to the random selection of labels in the case where more than one label has the same maximum number among neighboring nodes. Another drawback of LPA is the formation of a "monster" community. When community labels in a large community stop spreading, while labels in a small community near it have not yet spread, the labels of the large community will

likely affect the labels of the small community, and the small community will be swallowed. After several rounds of such phenomenon, a monster community may be formed (2007).

In view of these challenges, improvement of the initial label assignment, how ties are broken, updating and propagation strategies of labels in LPA were assigned in several approaches (Leung et al. 2009; Šubelj and Bajec 2011; Kothapalli et al. 2013; Lou et al. 2013; Xie and Szymanski 2013; Lin et al. 2014; Li et al. 2015). In order to avoid the birth of "monster" community with much smaller ones, Leung et al. (2009) proposed to modify the method by introducing a score for the labels, which decreases as the label propagates far from the vertex to which the label was originally assigned. They have suggested including label c_n into the maximal label consideration. When there are multiple maximal labels and one of them equals the concerned label c_n, the node retains its label. In this manner, a single label cannot span too large portions of the graph, and no monster communities can be recovered. From experimental results on a special structured network, they claimed that running time of the modified algorithm is $\mathcal{O}(\log n)$. However, nodes that are updated at the end of some iteration cannot efficiently propagate their final labels onward, as their neighbors have already been updated. On the other hand, a node that is considered first can possibly propagate its label to all of its neighbors, and thus form a community. For this, Šubelj and Bajec (2011) propose a balanced propagation that counteracts for the introduced randomness by utilizing node balancers. There are some improved algorithms, whose stability is enhanced, but complexity is increased. These algorithms lost the core advantage of LPA, which is nearly linear time complexity.

In a recent paper, Kothapalli et al. (2013) have designed a specific instance of LPA called Max-LPA (see Algorithm 7). It proceeds in rounds and in each round each node sends its label to all neighbors and then updates its label based on the labels received from neighbors and its own label (Denote by $l_v(t)$ the label of node u just before round t.). The algorithm can be described as a locally greedy coverage maximizer; i.e., it tries to maximize the fraction of edges which are placed within communities rather than across. With its purely local update rule, it tends to get stuck in local optima which implicitly are good solutions with respect to modularity. A label is likely to propagate through and cover a dense community, but unlikely to spread beyond bottlenecks (Staudt and Meyerhenke 2016). It has been analyzed on the Planted Partition Model for highly dense topologies. In particular, their analysis considers the static model $\mathcal{G}(n,p,q)$ with $p = \Omega(1/n^{1/4-\epsilon})$ and $q = \mathcal{O}(p^2)$ observe that in this case there are (with high probability) highly interconnected communities having constant diameter and a relatively small cut among them. In this very restricted case, they show the protocol converges in constant expected time and conjectured a logarithmic bound for sparse topologies.

Algorithm 7: Max-LPA on a node v

Data : A graph $G_0 = (V, E)$

$i = 0$

$l_v[i] \leftarrow random(0,1)$

while *true* **do**

  $i++$;

  send $l_v[i-1]$ to $\forall u \in N(v)$

  receive $l_u[i-1]$ from $\forall u \in N(v)$

  $l_v[i] \leftarrow max\{f \,|\, \sum_{u \in N'(v)}[l_u[i-1] == l] \geq \sum_{u \in N'(v)}[l_u[i-1] == l'] \text{ for all } l'\}$

end

The LPA is an almost linear time algorithm proved to be effective in finding a good community structure. For this, the multiresolution of communities can be produced by LPA. Firstly, Lou et al. (2013) adopt a measure named weighted coherent neighborhood propinquity in label update of LPA instead of the one that is shared by most of its neighbors. This measure is used to describe a quantity to evaluate the probability that a pair of vertices are involved in the same community. Similarly, Lin et al. (2014) propose an appropriate corresponding weight to each node in the application of LPA to detect communities. The node weight is assigned based on the community kernel. The community kernel is a set of nodes with greater influence than the other nodes in a network. Recently, Li et al. (2015) introduced a novel strategy to update the label synchronously by probability of each surrounding label to avoid oscillation and ensure the convergence of the algorithm. In this way, *LPA-S* can be easily parallelized. Buzun et al. (2014) propose a new LPA-based approach for community detection in social networks. Initially, they identify each user egomunities. The latter are cohesive subgroups within ego-network, the network of the user's closest neighbors (e.g., friends or followers). Then, all users are then initialized with community labels and start to iteratively exchange labels. During the iterations, communities are utilized to aggregate labels propagating to each user from his contacts. Finally, communities are post-processed so that poorly connected communities are split into smaller ones. LabelRank (2013) is a model based on operators to stabilize and boost the LPA, which avoid random output and improve the performance of community detection. Here, it uses node id's as labels. LabelRank stores, propagates and ranks labels in each node. During LabelRank execution, each node keeps multiple labels received from its neighbors. The running time of LabelRank is $\mathcal{O}(m)$.

5.1.2 Self-aggregation and self-organization

A self-organized approach is based on the assumption that each individual nodes are independently responsible for

determining the community to which they belong. The mechanism for making this decision is derived from Shannon entropy (2001) and requires a node to have knowledge only of its immediate neighbors. An application of the entropy introduced a new concept of node entropy in Collingsworth and Menezes (2014). With node entropy, each node is responsible for determining the community to which it belongs. The concept of a node entropy encapsulates the certainty of each node is regard to its current community. Otherwise, node entropy is a mathematical expression of an individual node's satisfaction with its current community. Its main idea was inspired by swarm intelligent systems, where each individual in the system interacts only with its immediate neighborhood, thus obtaining high efficiency and robustness. At each iteration, if a node has neighbors in different communities it calculates the entropy of joining the communities of its neighbors and has a higher probability of joining that having a lower entropy. This step is repeated until the entropy of the entire network stabilizes. A major advantage is the decentralized design of the algorithm which does not require extensive recalculation of the communities in case of a new node joining the network or a node leaving the network and is that all the changes are made locally. An automatic organization of a network into dense subgraphs according to the topological connection of each node has been proposed in Li et al. (2010). By applying neural networks, n input neurons corresponding to the nodes in complex network G and the output neurons represents c putative communities. A new operation and a new weight-updating scheme have been used to classify all nodes into no more than c groups according to their final winner neurons. The idea of another group of methods is the application of a genetic algorithm for community detection in complex networks (Halalai et al. 2010; Liu and Zeng 2010; Li et al. 2013; Said et al. 2018). In Halalai et al. (2010), the genetic algorithm uses the locus-based adjacency representation. Social networks are modeled by undirected graphs. According to this graph-based representation, an individual of the population consists of n genes, for a graph with n nodes. Each gene has a corresponding value from the range $\{1,..., n\}$. In the graph modeling the social network, a value i assigned to a gene j represents a link between nodes i and j. Thus, an individual of the population represents a partial graph with the same number of nodes as the original graph and at most N of its edges. Nodes that are connected in the partial graph will be assigned to the same component. A similar method has been proposed in Ghaemmaghami and Sarhadi (2013). Ren et al. (2011) use spatial and temporal characteristics in human's mobility to detect communities in a distributed manner. An interesting approach has been proposed by Hui et al. (2007) to distributed community detection scheme for Pocket Switched Networks. Community detection is seen as a local process taking place in each mobile device senses and detects its own local community by analyzing the history of mobile devices it has encountered. Only encounter events are used to construct the social relationships between users. Recently, an iterative method based on genetic algorithm (CC-GA) is proposed by Said et al. (2018) for detecting communities in social networks. Initially, CC-GA chooses one neighbor for each node. When choosing neighbors, CC-GA considers effective connectivity among nodes. Two nodes, v_i and v_j , are connected only if they are neighbors. In case a node has multiple neighbors, CC-GA chooses the neighbor with the highest Clustering Coefficient (CC). The CC is given as:

$$C_{V_i} = 2 \times \frac{L_i}{K_i(K_i - 1)} \tag{13}$$

where K_i the degree of the node V_i and L_i is the total number of links among neighbors of V_i, excluding the node itself. For nodes with a degree of one, there is no choice of best neighbor (based on CC), and therefore, the node is connected to the only one. The main objective of choosing the CC is to generate a population in such a way that densely connected components of the network are divided into separate communities. This algorithm produces better accuracy compared to other methods that have been developed based on the genetic algorithm.

5.1.3 Multiagent-based decentralized detection

Multiagent system is defined as a system consisting of a set of agents who operate in an environment and have the opportunity to interact. It aims to provide a general computing paradigm for solving hard computational problems as well as for characterizing complex systems. The important features of agents are, autonomy, self-aggregation, self-organization, and emergent behavior. Due to its inherent decentralized features, the multiagent system is especially suitable for solving large-scale and distributed problems such as distributed community detection. Recent reviews on community detection with distributed networks are presented in Danon et al. Yang et al. (2009). In this paper, we focused on decentralized community detection in undistributed network. Firstly, Yang and Liu (2007) proposed to dispatch a group of special computational agents into a network to collect information for community extraction. They autonomously collect a local view of the network and partition the complete network into communities. In their approach, each network node is modeled as one active agent (or a group of nodes managed by an active agent), which is responsible for gathering and processing local information through local interactions with others. In order to synchronize the collaborations among agents, they adopted a distributed clock mechanism, in which each agent maintains

its own clock in a decentralized way. Through a positive feedback mechanism, agents can self-organize them into desired network communities. To avoid synchronization, Huang et al. (2013) propose a novel decentralized algorithm based on a group of mobile agents that will be generated and dispatched into networks. Agents run concurrently and asynchronously without any synchronization constraints. They work together through a self-aggregation mechanism rather than direct P2P communication. Recently, a multiagent evolutionary method for discovering communities has been proposed in Ji et al. (2016). The focus of the method lies in the evolutionary process of computational agents in a lattice environment, where each agent corresponds to a candidate solution to the community detection problem. First, the method uses a connection-based encoding scheme to model an agent and a random-walk behavior to construct a solution. Next, it applies three evolutionary operators; i.e., competition, crossover, and mutation, to realize information exchange among agents and solution evolution. Bu et al. (2016) proposed a novel autonomy-oriented computing-based method for community mining (AOCCM) from the multiagent perspective. It utilizes reactive agents to pick the neighborhood node with the largest structural similarity as the candidate node, and thus determine whether it should be added into local community based on the modularity gain. They further improved AOCCM to a more efficient incremental version named AOCCM-i for mining communities from dynamic networks. The limitation of this work lies in that the selection of core nodes is based on the high degree.

5.2 Distributed community detection based on structural and semantic partition

The work in Gargi et al. (2011) studies this problem on the YouTube online video community and addresses challenging issues of working with a very large graph in the real-world. They consider the YouTube graph where each video represents a vertex and the edge between vertices captures their similarity. It must obtain the groups that cover a major portion of the graph by selecting the most important videos in the graph(seed video). The evaluation of the importance of a video is made by its popularity, which is computed on the basis of statistics such as the number of views, number of comments, etc. After selection of seed videos in pretreatment, a local partitioning algorithm is executed in parallel on each of the seed video. The last stage is the post-treatment, which divides the group into subgroups strongly linked to the inside as opposed to outside. Using descriptive video, authors calculate the occurrence of the most popular words associated with each video in the group. They also execute an iterative algorithm which removes small groups and combine strongly overlapped ones. At the end, detected groups have a high consistency of text, each corresponding to a different topic with a small overlap and high coverage of the entire YouTube graph.

Zhao et al. (2012) propose a topic-oriented community detection approach which combines social objects clustering and link analysis. Firstly, all social objects are clustered into different topics. Then, the members involved in those social objects are divided into different topical clusters, each corresponding to a certain topic. Finally, the link-based community detection for each topical cluster is performed to differentiate the strength of connections between members.

The ontology is introduced in Nayak and Biswas (2014) for finding the semantic community. The first step is the collection of data and the storage in the database, then the characteristics and properties retrieved from the database are represented in the form of an ontology. They consider that in the network every two nodes linked together form a community. These communities are combined with the previously stored ontology to form a modified ontology which will be used to find the degree of overlap for each feature included in ontology. These degrees of overlap are compared to finding a feature at a minimum value thus influencing group training.

Authors in Cruz et al. (2011) propose a joint between the semantic information from the social network represented by points of view, and structural information. It allows the combination of explicit social relations, the edges of the social graph, and implicit relationships called semantic. By using this information, it is possible to guide the graph-clustering process by adding information related to the similarity of the nodes in a real context. For this, community detection is divided into two phases. The first one consists in clustering the point of view using Kohonen maps (Kohonen et al. 1997) to obtain groups based on the similarity of the node features. Then the groups are used to change the weight of the edges in the graph, next in the second phase, a classic community detection algorithm is used.

In real-world scenarios, the model EILPA of Azaouzi and Romdhane (2017) introduces the social influence, as the semantic information, and the label propagation algorithm based on evidence theory for distributed community detection in static social networks. EILPA is mainly composed of two phases: (1) a seeding phase in which an optimal set of leaders is computed using the algorithm SAIM (Azaouzi and Romdhane 2018); and (2) an extension phase in which communities are extended from the leading nodes using a modified LPA (2007). The main idea of this second step is to extend the LPA algorithm by harnessing the influence measure of each node and the effect of the most influential nodes on the computed communities. The extension process is guided by measures of influence on the k-nearest neighbor rule. Doing it this way, EILPA becomes more stable compared to LPA and shows its efficiency on synthetic and real-world social networks.

6 Distributed community detection in dynamic social networks

6.1 Distributed community detection based on structural partition

Social networks offer a dominant abstraction of the organization and dynamics of varied kinds of people or people-to-technology interaction. However, they are large sizes with large changes over time. In this new communication paradigm, it is required that community detection is performed in a fully distributed way in these dynamic networks. It turns out nearly the same families of the above-discussed decentralized algorithms in static networks that are suitable for community detection in dynamic networks. Starting by the first family, Galluzzi (2012) proposed a distributed detection in dynamic networks by LP algorithm. It is based on the fact that if node i interacts (shares edges) mostly with members of community C then it is probably a member of community C during that time. If individuals are known to change community infrequently then the current community association of the individual is likely to be predictive of future community association. Another paper which applies an extension built upon LabelRank (Xie and Szymanski 2013) which is presented before time in Sect. 5.1.1 has been introduced in Xie et al. (2013) to distributed community detection in dynamic networks. The main idea is to adjust the detection as the network structure changes. Take advantage of what we already obtained in previous snapshot for inferring the dynamics in the current time step. Since local structure information is encoded in the node label distributions, the evolution of communities is expected to be caught and reflected in these distributions. The running time of the LabelRankT is the same as the original algorithm LabelRank, as it is $\mathcal{O}(m)$, linear with the number of edges. The space complexity is $\mathcal{O}(n)$ in practice because the number of labels in each node monotonically decreases and drops to a small constant in a few steps of the algorithm. Recently, an important work relying on the epidemic mechanism or Label Propagation is introduced by Clementi et al. (2015) in order to solve the problem of distributed detection of unknown communities in dynamic random graphs. Dynamic networks studied in this work are the Opportunistic Networks class as Intermittently Connected Mobile Networks (ICMNs) (2011). An application of the dynamic version of the well-studied Planted Bisection Model (Boppana 1987) dyn-$\mathcal{G}(n, p, q)$ which consists of a dynamic graph where n is the number of nodes while $p = p(n)$ and $q = q(n)$ are the edge-probability functions. The node set of the network is partitioned into two unknown communities and, at every time step, each possible $edge(u, v)$ is active with probability p if both nodes belong to the same community, while it is active with probability q (with $q \ll p$) otherwise. The core of this work is the combination between the labeling process of LPA and the dynamic graph as a finite-state Markov process. This combination is achieved by the application of local rules depending on the current node's neighborhood and on the current time step only. It is proven that when the ratio p / q is larger than n^b (for an arbitrarily small constant $b > 0$), the protocol finds the right "planted" partition in $\mathcal{O}(\log\ n)$ time even when the snapshots of the dynamic graph are sparse and disconnected (i.e., when $p = \Theta(1/n)$). Cordeiro et al. (2016) addressed the dynamic community detection problem in dynamic networks. The proposed method is a modification of the original Louvain algorithm. The distributed detection is described by a greedy optimization method that attempts to optimize the modularity of a partition of the network in successive iterations. Communities are calculated by maximizing the objective function in a two-step optimization in each one of the iterations. In the first step, small communities are formed by optimizing the modularity locally. In the following step, nodes belonging to the same community are aggregated in a single node that represents a community in a new aggregated network of communities. Iteratively these steps are repeated until no increase in modularity is possible with a hierarchy of communities being produced.

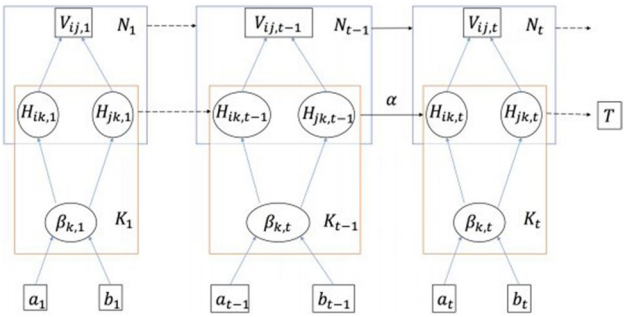

Fig. 10 Graph model of DBNMF. Here $V_{ij,t}$ represent the element at the $i-th$ rows $j-th$ column on the temporal network at snapshot t, and $H_{ik,t}$ denotes the expected probability that node i belongs to community k at snapshot t

With the dynamic network, four types of operations to communities were defined when nodes are added or removed from the network: The algorithm maintains unchanged all the communities that were not affected by modifications to the network. Else, the algorithm performs a local modularity optimization that maximizes the modularity gain function only for those communities where the editing of nodes and edges was performed. The algorithm merges two communities into one or assigns nodes to an existing community or creates a new community with new nodes.

In Wang et al. (2016a), The authors proposed a Bayesian probabilistic model, named Dynamic Bayesian Nonnegative Matrix Factorization (DBNMF), for automatic detection of

overlapping communities in temporal networks (see Fig. 10). In the DBNMF model, the community structure at snapshot $t+1$ is influenced by the community structure of the snapshot t and independent of the previous snapshot networks. For each snapshot network, DBNMF tied the columns of the membership matrix through scale parameters that are drawn from Half-Normal distribution, the smaller the parameter the less weight of the column. After finishing all the calculations of the temporal network, the columns whose weights are close to zero in the membership matrix are removed. Then, the overlapping community results and the number of communities are derived simultaneously. The algorithm has complexity of $O((T_p)(eK_{\text{initial}}))$, where p is the average number of the iterations at snapshot networks t and e denotes the average edges for all snapshots of the temporal network and K_{initial} the number maximal of communities in each snapshot of the temporal network. Besides, the proposed model can be applied to large and sparse networks because of its stability and effectiveness, which is partly validated in the experiments.

Due to the execution time and the big size of data in each time slot for the dynamic network, the independent investigation of each time slot is not efficient for communities. For this, Asadi and Ghaderi (2018) proposed an unsupervised machine learning algorithm, called Incremental Speaker-Listener Propagation Algorithm (ISLPA). ISLPA can detect communities incrementally after removing or adding a batch of nodes and edges over time. The general idea of the proposed method is to use the information obtained from the graph's analysis at time slot $t-1$ to improve the analysis at time slot t.

6.2 Distributed community detection based on structural and semantic partition

Huang and Yang (2012) proposed an approach based on semantic clustering. This approach consists of 3 parts; the first step is to initialize the communities through the fuzzy classification using the measure of semantic similarity applied to post representations to find communities in a social network. Only post-community relations with no similarity values below a threshold are kept, and the representation of every community is created from the integration of representations of the messages that have relationships with the community. The second step is the dynamic detection of communities, which occurs when new messages are received while creating its representation. Then, they compute the semantic similarity with each community in the social network if there is an active community whose representation similarity is equal to or greater than the threshold with the representation of the new post. The relation "belonging to" between the community and the post is built. Otherwise, if not the similarity measure is performed with the representations outdated communities for the new post. If there is no active community or outdated for the new post, then a new community is built. The third step is the community outdated Booking (Renaissance/Remove). It is necessary that the number of new post of each examined community comes to update its representation which must reach a predefined threshold for a predefined time interval. Otherwise withdraw from social network and reserved as outdated. Any outdated community can be active once again when there is a sufficient number of new posts during a defined time interval, if not, the community and its messages will be discarded.

A framework that considers a set of users U and a set of resources R at a given site (music files, videos, etc.) is presented in Abrouk et al. (2010). They assume that users emit a vote on a subset of resources site. These votes are illustrated by a matrix $M : |U| * |R|$ defined as follows:

$$M(u_i, r_j) = \begin{cases} 1, & \text{if } u_i \text{ has the interest for } r_j \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

This matrix is dynamically updated when new users, new resources or new uses appear. At first they assume that a set of tags T is set and each resource is annotated with a subset of these tags. The proposed approach allows to divide users into different communities, based on tags groups that they appreciate. For this, they propose calculating the degree of membership $x_{i,j}$ of user u_i to a tag t_j as:

$$x_{i,j} = \frac{|A_{u_i,t_j}|}{|A_{u_i}|} \tag{15}$$

The more the coefficient $x_{i,j}$ is close to 1, the more the user i manipulates tags type j. The second step is to bring together similar tags, statistically with the technique of principal component analysis (PCA). This step aims to find linear combinations of the variables representing the tags to best explain the interests of users in order to provide relevant main components for the analysis of uses, ignoring the tags for each component located in the low correlation zone and collecting the tags located in areas of high correlation. Once all the tags T are decomposed into K tags communities, we deduce user communities. We look at the majority community tag for each user through calculating the degree of membership of each user to each tag community. This method has the advantage of reducing the number of tags manipulated, which allows to treat large size networks.

Cattuto et al. (2008) present another statistical approach to evaluate the semantic distances. To construct weighted network resources the authors use the annotation data. In this context, the similarity between resources is proportional to the overlap of their tags games. They use a weighted method named TF-IDF (Term Frequency-Inverse Document

Table 3 Summary table of distributed community detection

Method	Detection nature	Network nature	Predefined community number	Overlap	Complexity	Advantages	Drawbacks
LP (2007)	Structural	Static	No	Yes	$\mathcal{O}(n)$	Simplicity of detection Parallelized or distributed process	Formation of a monster community Convergence problem Unstable because of randomness Weak robustness
BP (2011)	Structural	Static	No	No	$\mathcal{O}(n)$	More robust Relatively stableA	–
Leung et al. (2009)	structural	static	No	No	$\mathcal{O}(\log n)$	No monster communities	–
CK-LPA (2014)	Structural	Static	No	No	$\mathcal{O}(n+m)$	The convergence time is reduced Effectively used in large Scale social networks	–
LPA-CNP (2013)	Structural	Static	No	No	$\mathcal{O}(k.(n+m).d^2_{max})$ $\mathcal{O}(n)$ if the network is sparse $d_{\max}$ is the maximum degree of nodes	More robust and stable than LPA More effective in performance	–
LabelRank (2013)	Structural	Static	No	No	$\mathcal{O}(m)$	Stable and good detection quality	–
LPA-S (2015)	Structural	Static	No	Yes	$\mathcal{O}(km)$ k: the number of iteration	Easily parallelized Better performance	–
Max-LPA (2013)	Structural	Static	No	Yes	$\mathcal{O}(\log n)$	Fatly and correctly converges	–
Collingsworth and Menezes (2014)	Structural	Static	No	No	$\mathcal{O}(n)$	A self-organizing community detection Successfully identify a high-resolution partitioning Suited to manage dynamically changing networks.	–
SOM (2010)	Structural	Static	No	No	$\mathcal{O}(n)$	Has no resolution The efficiency and effectiveness on both weighted and undirected networks Suitable for large-scale heterogeneous	–
MENSGA (2013)	Structural	Static	No	No	–	High accuracy	–
Ren et al. (2011)	Structural	Static	No	No	–	High detection rate Low false positive rate	–

Table 3 (continued)

Method	Detection nature	Network nature	Predefined community number	Overlap	Complexity	Advantages	Drawbacks
Gargi et al. (2011)	Structural and semantic	Static	No	No	–	Large-scale Parallel fashion Cover large portions of the graph	Data loss
Zhao et al. (2012)	Structural and Semantic	Static	Yes	No	$\mathcal{O}(h * m * n * k)$ h: the number of iterations m: the number of social objects n: dimensions of social objects k: the number of clusters	Applied to many kinds of social networks Improve the efficiency of collaborative learning	Low performance when the link is more important than the topic
Nayak and Biswas (2014)	Structural and semantic	Static	No	Yes	–	Reuse of stored ontology for different algorithms Updated whenever needed	Data loss
Cruz et al. (2011)	Structural and semantic	Static	No	No	$\mathcal{O}(\lvert F_v^*\rvert^3 \rvert * V)$ $\lvert F_v^*\rvert$: the number of features in the point of view – V: the non-empty set of vertices	Better performance according to the modularity and the average intra-cluster distance	High time complexity
Huang and Yang (2012)	Structural and semantic	Dynamic	No	No	–	Help users to exchange opinions which they are interested	–
Abrouk et al. (2010)	Structural and semantic	Dynamic	No	No	–	Relevant and reliable results Eliminate spam Reduce the number of tags manipulated	– High cost
Cattuto et al. Cattuto et al. (2008)	Structural and semantic	Dynamic	No	Yes	–	Meaningful social classification of resources	It does not reduce the number of manipulated tags

Table 4 Summary of performance over real-world networks

Category	Dataset	Algorithm	Ref.	NMI value	Q value
C4	Football (Girvan and Newman 2002)	LabelRank	Xie and Szymanski (2013)	0.5908	0.60
C3	Facebook	AOCCMNO	Bu et al. (2016)	0.930	0.73
C1	Facebook		Fan and Yeung (2014)	0.865	0.7011
C4	Citation network dataset	LabelRank	Cordeiro et al. (2016)	0.620	0.59
C1	B3	MeanCD	Chaabani and Akaichi (2017)	0.620	0.67
C3	Football		Huang et al. (2013)	0.5908	0.91
C2	Arxiv	DSLM	Huang et al. (2013)	Unchanged	Unchanged

Frequency) (Salton and McGill 1986) which allows the evaluation of the importance of a term contained in a document, the weight increases with the number of occurrences of the word in the document. It is used in this approach to take into account the representativeness of tags to detect user communities by the similarities of their tags. They then use the Pearson correlation coefficient as a measure of similarity and apply partitioning methods. The disadvantage of this method is that it does not reduce the number of manipulated tags, which may be extremely large. In Nath and Roy (2018) a new incremental parallel community detection method, *PcDEN* (Parallel Community Detection approach in Evolving Networks) is proposed. PcDEN can detect communities in dynamic distributed networks. They define a new affinity score based on intra-community strength between nodes and their neighbors. They also derive a new model to perform community merging, based on common high degree nodes present in both the communities.

The second part of this survey which contains the different decentralized communities detection methods is terminated by an overview Table 3.

7 Comparison

The efficiency of any community detection algorithm can be determined stated categorically with their working principles and implementations. In this section, we furnish different metrics used for evaluating the performance, the features of few standard benchmark networks used by different community detection algorithms in the literature.

To facilitate the understanding of readers, a comparison of different algorithms based on the measurement of important quality measures such as normalized mutual information, etc., is given in Table 4.

7.1 Metrics

In the literature (Fortunato 2010), the quality of communities in real-world networks is determined on the basis of different measures which are widespread throughout in the literature. A brief description of some of those measures is as follows.

Modularity Q measures the goodness of the division of the network with respect to the whole network. A high value of modularity indicates the good partitioning of the network. The Newman's modularity is used to measure the strength of the divisions of the network into partitions and is given as Q_p (Fortunato 2010).

$$Q_p = \frac{1}{2m}\sum_{i,j}\left(A_{ij} - \frac{d_i d_j}{2m}\right)\delta(C_i, C_j) \tag{16}$$

where A is the adjacency matrix, m is the total number of edges, d_i is the degree of the node i, and $\delta(C_i, C_j) = 1$ if node i and node j belong to the same community, otherwise $\delta(C_i, C_j) = 0$.

Normalized Mutual Information measure (NMI) is a measure of similarity between the partitions based on the information theory.

$$\mathrm{NMI}(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} N_{ij}\log\left(\frac{N_{ij}N}{N_{i_s}N_{j_s}}\right)}{\sum_{i=1}^{C_A} N_{i_s}\log\left(\frac{N_{i_s}}{N}\right) + \sum_{j=1}^{C_B} N_{j_s}\log\left(\frac{N_{j_s}}{N}\right)} \tag{17}$$

where N is the confusion matrix in which rows correspond to the 'real' communities and columns correspond to the 'observed' communities. N_{ij} is the number of nodes in the real community i that appear in the observed community j. C_A and C_B represent the number of real communities and observed communities, respectively. N_{i_s} and N_{j_s} represent the sum over the ith row of matrix N and jth column of the matrix N, respectively. The value of NMI ranges between 0 and 1. NMI index has a value 1 if the estimated community matches the reference community and 0 in the opposite case.

7.2 Benchmarks

The best way to classify the relative performance of these algorithms is to test them on two types of network datasets: real-world network datasets and synthetic network datasets. The real-world networks are network instances representing some real-world applications. Table 4 shows some real-world data sets[7] of different networks having known community structure. Unfortunately, in most of the method, the evaluation on the synthetic networks is missing. For the ease of readers, we note the category of each method, C1, C2, C3, C4 denote centralized detection in static network, centralized detection in dynamic network, distributed detection in static network and distributed detection in dynamic network, respectively. The analysis of the performance of the community detection algorithms is based on multiple parameters like complexity, the computational cost and requirement of the main memory to implement the particular method. The first two aspects described computational cost of each algorithm, is stated in Tables 3. The NMI scores of most approaches in dynamic networks are fall short and are far from stable, while those obtained in static networks are very high and relatively close to 1. Obviously, the main limit of community detection in dynamic social networks is the instability. Then, it is impossible to distinguish between

[7] http://snap.stanford.edu/data/index.html/, Apr. 2019.

changes due to the evolution of the community structure and changes due to the instability of algorithms. The other bottleneck in community detection is the memory consumption. After that comparative study, we noticed that complexity of the algorithm determines the effect of the growth of cost of the algorithm over network size and dynamic. Finally, a comparison on running time on these approaches shows that time taken for centralized detection is much more than distributed detection.

8 Applications of community detection in social networks

This section discusses the application of community detection for social networks. In Online Social Networks, Yang et al. (2010) discussed applications of community mining algorithms in social networks that range from network reduction for large-scale network analysis to community mining in distributed and dynamic networks. In another work, Java et al. (2007) studied topological and geographical properties of Twitter's social network and presented the observations of the microblogging process. As the social networks Flickr, the community detection algorithm to extract groups of related images within large image collections is defined in Gu et al. (2015). Gkini and Brailas (2015) have studied large-scale community structures in Facebook network.

In Communication Networks, the community structure has enabled a wide variety of applications such as efficient routing in Mobile Ad Hoc Networks (MANETs) and worm containment in Online Social Networks . Considering the dynamic of MANETs, these algorithms are able to recompute of the network structure at every change in the network topology with reasonable processing time and computational costs. For example, Blondel et al. (2008a) studied a mobile phone communication network and applied a fast hierarchical modularity optimization algorithm proposed by the authors themselves.

In E-Commerce, detecting communities enhances enhancing online shoppers, recommended products and targeted advertising. In this context, the paper (Gasparetti et al. 2017) presents a survey of previous studies done on community detection and recommender systems.

In Academia and Scientometrics, the community detection plays a vital role in academics, i.e., it is very important to monitor and predict the performance of students. For example, Kosmides et al. (2014) proposed a novel approach for robust and accurate estimation of research collaborations exploiting the innovative concept of Virtual Communities.

In Biological Systems and Healthcare, There are a variety of community detection algorithms developed for social networks that can also be successfully extended to the biological networks. For example, the community detection is used in Calderone et al. (2016) to compare Alzheimer's and Parkinson's interaction networks.

9 Future directions and open challenges

After the thorough study of the existing models and algorithms for community detection in social networks, we provide the possible research directions in future. This section discusses the challenging problems and future research directions that are still awaiting the solution.

Dynamic semantic data: Most of the proposed methods are very good to detect the communities in dynamic structure. One prominent hurdle faced by existing clustering algorithms is dealing with dynamic semantic data. As shown in Sect. 6.2, only few available community detection based on structural and semantic partition algorithms can handle the large dynamic networks. Therefore, there is demand for community detection algorithms that are scalable for large networks.

Scalability of Community Detection Algorithms: The dimensionality reduction in case of big data is the key concern in modern algorithms which tend to bring the data to low dimensional space before discovering patterns. In fact, this technique is named "graph's reduction" and represents a significant task in social networks' analysis. These algorithms suffer from risk of losing key characteristics of communities. Hence, there is still need to make algorithms more efficient, scalable and distributed.

Estimation of Number of Communities in Dynamic Networks: To understand the structural and functional properties of these time-varying networks, it is desirable to detect and analyze the evolving community structure. In temporal networks, the identified communities should reflect the current snapshot network, and the same time be similar to the communities identified in history or the previous snapshot networks. Most of the existing approaches assume that the number of communities is known or not heuristic methods. This is unsuitable and complicated for most real-world networks, especially temporal networks (Wang et al. 2016a).

Distributed Detection Community on undistributed Social Network: With large-scale networks, the most of proposed methods aims to distribute detection framework with shared-memory parallelism. The main obstacle in this distributed detection framework lies in the fact that the community mining algorithms often exhibit a high degree of data dependency. To address this issue, a new category is presented, in which several communities are detected in a distributed manner using an undistributed social graph (Azaouzi and Romdhane 2017). Therefore, there is a need for distributed detection algorithms that can help in undistributed social networks.

Multimodal social network analysis: Although information on social media can be of different modalities such as texts, images, audio or videos, traditional approaches usually represent only one prominent modality. Therefore, there is a need for heuristic algorithms that can help in combining information from different modalities to classify social media content and are able to handle the above problems with existing techniques.

Dynamic Multilayer Networks: One way to represent the social networks is as multilayer graphs, where each layer contains a unique set of edges over the same underlying vertices. Edges in different layers typically have related but distinct semantics. For example, the friend relationships might complement behavioral measures that link users according to their actions or interests. Modeling the evolution and detecting the efficient communities in dynamic multilayer networks can be considered as a good research problem.

10 Conclusions

In this survey, we made a thorough review of available community detection algorithms for static/dynamic social networks. The main goal of this article is to categorize the community detection models, to study their characteristics, and to analyze their performance on real-world network datasets. The comparative study reveals the fact that distributed community detection algorithms are performing well compared to the state-of-the-art algorithms. This will create interest to the researchers to work more in this domain. Some possible further research directions and open challenges in community detection have been addressed.

References

Abrouk L, Gross-Amblard D, Leprovost D (2010) Decouverte de communautes par analyse des usages. extraction et gestion des connaissances-Atelier Web Social A5–5

Aktunc R, Toroslu IH, Ozer M, Davulcu H (2015) A dynamic modularity based community detection algorithm for large-scale networks: Dslm. In: Proceedings of the 2015 IEEE/ACM international conference on advances in social networks analysis and mining, ASONAM'15, pp 1177–1183

Albert R, Barabási A-L (2002) Statistical mechanics of complex networks. Rev Mod Phys 74(1):47–97

Asadi M, Ghaderi F (2018) Incremental community detection in social networks using label propagation method. In: Proceedings of the 2018 23rd conference of open innovations association, FRUCT'18, pp 39–47

Azaouzi M, Romdhane LB (2017) An evidential influence-based label propagation algorithm for distributed community detection in social networks. Procedia Computer Science, 112:407 – 416. In: Proceedings of the 21st international conference on knowledge-based and intelligent information and engineering systems, KES2017, 6–8 Sep 2017, Marseille, France

Azaouzi M, Romdhane LB (2018) An efficient two-phase model for computing influential nodes in social networks using social actions. J Comput Sci Technol 33(2):286–304

Barnard ST, Simon HD (1994) Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. Concurr Comput Pract Exp 6(2):101–117

Ben Romdhane L, Chaabani Y, Zardi H (2013) A robust ant colony optimization-based algorithm for community mining in large scale oriented social graphs. Expert Syst Appl 40(14):5709–5718

Bezdek JC, Ehrlich R, Full W (1984) Fcm: the fuzzy c-means clustering algorithm. Comput Geosci 10(2–3):191–203

Bhat SY, Abulaish M (2015) Hoctracker: tracking the evolution of hierarchical and overlapping communities in dynamic social networks. IEEE Trans Knowl Data Eng 27(4):1013–1019

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008a) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):P10008

Boppana RB (1987) Eigenvalues and graph bisection: an average-case analysis. In: 28th annual symposium on foundations of computer science, pp 280–285

Bu Z, Wu Z, Cao J, Jiang Y (2016) Local community mining on distributed and dynamic networks from a multiagent perspective. IEEE Trans Cybern 46(4):986–999

Buzun N, Korshunov A, Avanesov V, Filonenko I, Kozlov I, Turdakov D, Kim H (2014) Egolp: fast and distributed community detection in billion-node social networks. In: Proceedings of the 2014 IEEE international conference on data mining workshop, ICDMW'14, pp 533–540

Calderone A, Formenti M, Aprea F, Papa M, Alberghina L, Colangelo AM, Bertolazzi P (2016) Comparing alzheimer's and parkinson's diseases networks using graph communities structure. BMC Syst Biol 10(1):25

Cattuto C, Baldassarri A, Servedio VD, Loreto V (2008) Emergent community structure in social tagging systems. Adv Complex Syst 11(04):597–608

Chaabani Y, Akaichi J (2017) Meaningful communities detection in medias network. Soc Netw Anal Min 7(1):1–11

Chakraborty T, Dalmia A, Mukherjee A, Ganguly N (2017) Metrics for community analysis: a survey. ACM Comput Surv 50(4):54:1–54:37

Charikar MS (2002) Similarity estimation techniques from rounding algorithms. In: Proceedings of the thiry-fourth annual ACM symposium on theory of computing, STOC'02, pp 380–388

Chevalier C, Safro I (2009) Comparison of coarsening schemes for multilevel graph partitioning. In: Learning and intelligent optimization, pp 191–205

Clementi A, Di Ianni M, Gambosi G, Natale E, Silvestri R (2015) Distributed community detection in dynamic graphs. Theor Comput Sci 584:19–41

Collingsworth B, Menezes R (2014) A self-organized approach for detecting communities in networks. Soc Netw Anal Min 4(1):169

Cordeiro M, Sarmento RP, Gama J (2016) Dynamic community detection in evolving networks using locality modularity optimization. Soc Netw Anal Min 6(1):15

Costa LF, Oliveira ON Jr, Travieso G, Rodrigues FA, Villas Boas PR, Antiqueira L, Viana MP, CorreaRocha LE (2011) Analyzing and modeling real-world phenomena with complex networks: a survey of applications. Adv Phys 60(3):329–412

Cruz JD, Bothorel C, Poulet F (2011) Semantic clustering of social networks using points of view. CORIA: conférence en recherche d'information et applications. Avignon, France, pp 175–182

Dang T, Viennet E (2012) Community detection based on structural and attribute similarities. In: International conference on digital society, ICDS'12, pp 7–14

Dhillon IS, Guan Y, Kulis B (2007) Weighted graph cuts without eigenvectors: a multilevel approach. IEEE Transa Pattern Anal Mach Intell 29(11):1944–1957

Faloutsos M, Faloutsos P, Faloutsos C (1999) On power–law relationships of the internet topology. SIGCOMM Comput Commun Rev 29(4):251–262

Fan W, Yeung K (2014) Incorporating profile information in community detection for online social networks. Phys A Stat Mech Appl 405:226–234

Feng H, Tian J, Wang HJ, Li M (2015) Personalized recommendations based on time-weighted overlapping community detection. Inf Manag 52(7):789–800

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75–174

Galluzzi V (2012) Real time distributed community structure detection in dynamic networks. In: 2012 IEEE/ACM international conference on advances in social networks analysis and mining, ASONAM'12, pp 1236–1241

Gargi U, Lu W, Mirrokni VS, Yoon S (2011) Large-scale community detection on youtube for topic discovery and exploration. In: Proceedings of the fifth international conference on weblogs and social media, ICWSM'11, pp 486–489

Gasparetti F, Micarelli A, Sansonetti G (2017) Community detection and recommender systems. Springer, New York, pp 1–14

Ge R, Ester M, Gao BJ, Hu Z, Bhattacharya B, Ben-Moshe B (2008) Joint cluster analysis of attribute data and relationship data: the connected k-center problem, algorithms and applications. ACM Trans Knowl Discov Data 2(2):7:1–7:35

Ghaemmaghami F, Sarhadi RM (2013) Somsn: an effective self organizing map for clustering of social networks. Int J Comput Appl 84(5):7–12

Girvan M, Newman ME (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826

Gkini C, Brailas A (2015) Visualizations of personal social networks on facebook and community structure: an exploratory study. Eur J Soc Behav 2(1):21–30

Greene D, Doyle D, Cunningham P (2010) Tracking the evolution of communities in dynamic social networks. In: Proceedings of the 2010 international conference on advances in social networks analysis and mining, ASONAM'10, pp 176–183

Gregori E, Lenzini L, Mainardi S (2013) Parallel k-clique community detection on large-scale networks. IEEE Trans Parallel Distrib Syst 24(8):1651–1660

Gu Y, Qian X, Li Q, Wang M, Hong R, Tian Q (2015) Image annotation by latent community detection and multikernel learning. IEEE Trans Image Process 24(11):3450–3463

Halalai R, Lemnaru C, Potolea R (2010) Distributed community detection in social networks with genetic algorithms. In: Proceedings of the 2010 IEEE international conference on intelligent computer communication and processing, ICCP'10, pp 35–41

He J, Chen D (2015) A fast algorithm for community detection in temporal network. Phys A Stat Mech Appl 429:87–94

Hendrickson B, Leland R (1995) A multilevel algorithm for partitioning graphs. In: Proceedings of the 1995 ACM/IEEE conference on supercomputing, supercomputing'95, p 28

Hopcroft J, Khan O, Kulis B, Selman B (2004) Tracking evolving communities in large linked networks. Proc Natl Acad Sci 101(suppl 1):5249–5253

Hu P, Lau WC (2013) A survey and taxonomy of graph sampling. arXiv:1308.5865

Huang HH, Yang HC (2012) Semantic clustering-based community detection in an evolving social network. In: 2012 sixth international conference on genetic and evolutionary computing, ICGEC'12, pp 91–94

Huang J, Yang B, Jin D, Yang Y (2013) Decentralized mining social network communities with agents. Math Comput Model 57(11):2998–3008

Hübler C, Kriegel HP, Borgwardt K, Ghahramani Z (2008) Metropolis algorithms for representative subgraph sampling. In: Proceedings of the 2008 eighth IEEE international conference on data mining, ICDM '08, pp 283–292

Hui P, Yoneki E, Chan SY, Crowcroft J (2007) Distributed community detection in delay tolerant networks. In: Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture, MobiArch'07, pp 7

Java A, Song X, Finin T, Tseng B (2007) Why we twitter: understanding microblogging usage and communities. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis. ACM, pp 56–65

Ji J, Jiao L, Yang C, Liu J (2016) A multiagent evolutionary method for detecting communities in complex networks. Comput Intell 32(4):587–614

Kang U, Faloutsos C (2011) Beyond'caveman communities': hubs and spokes for graph compression and mining. In: Proceedings of the 2011 IEEE 11th international conference on data mining, ICDM'11, pp 300–309

Karypis G, Kumar V (1998a) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20(1):359–392

Karypis G, Kumar V (1998b) Multilevel k-way partitioning scheme for irregular graphs. J Parallel Distrib Comput 48(1):96–129

Kernighan BW, Lin S (1970) An efficient heuristic procedure for partitioning graphs. Bell Syst Tech J 49(2):291–307

Kohonen T, Kaski S, Lappalainen H (1997) Self-organized formation of various invariant-feature filters in the adaptive-subspace som. Neural Comput 9(6):1321–1344

Kosmides P, Adamopoulou E, Demestichas K, Remoundou C, Loumiotis I, Theologou M (2014) Community awareness in academic social networks. In: 2014 IEEE/ACM 7th international conference on utility and cloud computing, pp 647–651

Kothapalli K, Pemmaraju SV, Sardeshmukh V (2013) On the analysis of a label propagation algorithm for community detection. In: Proceedings of the 14th international conference on distributed computing and networking, ICDCN'13, pp 255–269

Krishnamurthy V, Faloutsos M, Chrobak M, Lao L, Cui J-H, Percus AG (2005) Reducing large internet topologies for faster simulations. Networking 5:328–341

Kuzmin K, Shah SY, Szymanski BK (2013) Parallel overlapping community detection with slpa. In: Proceedings of the 2013 international conference on social computing, SocialCom'13, pp 204–212

Lancichinetti A, Fortunato S, Kertész J (2009) Detecting the overlapping and hierarchical community structure in complex networks. New J Phys 11(3):033015

LaSalle D, Karypis G (2015) Multi-threaded modularity based graph clustering using the multilevel paradigm. J Parallel Distrib Comput 76:66–80

Leskovec J, Faloutsos C (2006) Sampling from large graphs. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'06, pp 631–636

Leung IX, Hui P, Lio P, Crowcroft J (2009) Towards real-time community detection in large networks. Phys Rev E 79(6):066107

Li S, Lou H, Jiang W, Tang J (2015) Detecting community structure via synchronous label propagation. Neurocomputing 151:1063–1075

Li Y, Liu G, Lao S-Y (2013) A genetic algorithm for community detection in complex networks. J Central South Univ 20(5):1269–1276

Li Z, Liu J, Wu K (2018) A multiobjective evolutionary algorithm based on structural and attribute similarities for community detection in attributed networks. IEEE Trans Cybern 48(7):1963–1976

Li Z, Wang R, Zhang X, Chen L (2010) Self-organizing map of complex networks for community detection. J Syst Sci Complex 23(5):931–941
Lin Y-R, Chi Y, Zhu S, Sundaram H, Tseng BL (2009) Analyzing communities and their evolutions in dynamic social networks. ACM Trans Knowl Discov Data 3(2):8
Lin Z, Zheng X, Xin N, Chen D (2014) Ck-lpa: Efficient community detection algorithm based on label propagation with community kernel. Phys A Stat Mech Appl 416:386–399
Liu J, Zeng J (2010) Community detection based on modularity density and genetic algorithm. In: Proceedings of the 2010 international conference on computational aspects of social networks, CASoN'10, pp 29–32
Lou H, Li S, Zhao Y (2013) Detecting community structure using label propagation with weighted coherent neighborhood propinquity. Phys A Stat Mech Appl 392(14):3095–3105
Lumsdaine A, Gregor D, Hendrickson B, Berry JW (2007) Challenges in parallel graph processing. Parallel Process Lett 17(1):5–20
Maiya AS, Berger-Wolf TY (2010) Sampling community structure. In: Proceedings of the 19th international conference on World wide web, WWW'10, pp 701–710
Mansour N, Ponnusamy R, Choudhary A, Fox GC (1993) Graph contraction for physical optimization methods: a quality-cost tradeoff for mapping data on parallel computers. In: Proceedings of the 7th international conference on supercomputing, ICS'93, pp 1–10
McDaid A, Hurley N (2010) Detecting highly overlapping communities with model-based overlapping seed expansion. In: Proceedings of the 2010 international conference on advances in social networks analysis and mining, ASONAM'10, pp 112–119
Milgram S (1967) The small world problem. Psychol Today 67(1):61–67
Nath K, Roy S (2018) A parallel approach to detect communities in evolving networks. In: Proceedings of the international conference on big data analytics, BDA'18, pp 188–203
Nayak V, Biswas B (2014) Finding prominent features in communities in social networks using ontology, pp 31–36
Neville J, Adler M, Jensen D (2003) Clustering relational data using attribute and link information. In: Proceedings of the text mining and link analysis workshop, 18th international joint conference on artificial intelligence, pp 9–15
Newman ME, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113
Nguyen NP, Dinh TN, Shen Y, Thai MT (2014) Dynamic social community detection and its applications. PLoS ONE 9(4):e91431
Noack A, Rotta R (2009) Multi-level algorithms for modularity clustering. SEA 9:257–268
Osborne F, Scavo G, Motta E (2014) A hybrid semantic approach to building dynamic maps of research communities. In: International conference on knowledge engineering and knowledge management, EKAW'15, pp 356–372
Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E 76(3):036106
Ren Y, Chuah MC, Yang J, Chen Y (2011) Distributed spatio-temporal social community detection leveraging template matching. In: Proceedings of the 2011 IEEE global telecommunications conference, GLOBECOM'11, pp 1–6
Rhouma D, Romdhane LB (2014) An efficient algorithm for community mining with overlap in social networks. Expert Syst Appl 41(9):4309–4321
Rhouma D, Romdhane LB (2018) An efficient multilevel scheme for coarsening large scale social networks. Appl Intell 48(10):3557–3576
Riedy EJ, Meyerhenke H, Ediger D, Bader DA (2011) Parallel community detection for massive graphs. In: Proceedings of the 9th international conference on parallel processing and applied mathematics, pp 286–296
Ruan Y, Fuhry D, Liang J, Wang Y, Parthasarathy S (2015) Community discovery: simple and scalable approaches. In: User community discovery. Springer, pp 23–54
Sadi S, Ögüdücü Ş, Uyar A. Ş (2010) An efficient community detection method using parallel clique-finding ants. In: Proceedings of the 2010 IEEE congress on evolutionary computation, CGC'10, pp 1–7
Safro I, Ron D, Brandt A (2009) Multilevel algorithms for linear ordering problems. JEA 13:4:1.4–4:1.20
Said A, Abbasi RA, Maqbool O, Daud A, Aljohani NR (2018) Cc-ga: a clustering coefficient based genetic algorithm for detecting communities in social networks. Appl Soft Comput 63:59–70
Salton G, McGill MJ (1986) Introduction to modern information retrieval. McGraw-Hill Inc, New York
Saltz M, Prat-Pérez A, Dominguez-Sal D (2015) Distributed community detection with the wcc metric. In: Proceedings of the 24th international conference on world wide web, WWW'15, pp 1095–1100
Samie ME, Hamzeh A (2018) Change-aware community detection approach for dynamic social networks. Appl Intell 48(1):78–96
Satuluri V, Parthasarathy S (2009) Scalable graph clustering using stochastic flows: applications to community discovery. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'09, pp 737–746
Satuluri V, Parthasarathy S, Ruan Y (2011) Local graph sparsification for scalable clustering. In: Proceedings of the 2011 ACM SIGMOD international conference on management of data, SIGMOD'11, pp 721–732
Shang J, Liu L, Li X, Xie F, Wu C (2016) Targeted revision: a learning-based approach for incremental community detection in dynamic networks. Phys A Stat Mech Appl 443:70–85
Shannon CE (2001) A mathematical theory of communication. ACM SIGMOBILE Mob Comput Commun Rev 5(1):3–55
Staudt CL, Meyerhenke H (2013) Engineering high-performance community detection heuristics for massive graphs. In: Proceedings of the 2013 42nd international conference on parallel processing, ICPP'13, pp 180–189
Staudt CL, Meyerhenke H (2016) Engineering parallel algorithms for community detection in massive networks. IEEE Trans Parallel Distrib Syst 27(1):171–184
Steinhaeuser K, Chawla NV (2008) Community detection in a large real-world social network. In: Liu H, Salerno JJ, Young MJ (eds) Social computing, behavioral modeling, and prediction. Springer, Boston, MA, pp 168–175
Šubelj L, Bajec M (2011) Robust network community detection using balanced propagation. Eur Phys J B Condens Matter Complex Syst 81(3):353–362
Waltman L, van Eck NJ (2013) A smart local moving algorithm for large-scale modularity-based community detection. Eur Phys J B 86(11):471
Wang T, Chen Y, Zhang Z, Xu T, Jin L, Hui P, Deng B, Li X (2011) Understanding graph sampling algorithms for social network analysis. In: 2011 31st international conference on distributed computing systems workshops, ICDCSW'11, pp 123–128
Wang C-D, Lai J-H, Philip SY (2014a) Neiwalk: community discovery in dynamic content-based networks. IEEE Trans Knowl Data Eng 26(7):1734–1748
Wang Z, Zhang D, Zhou X, Yang D, Yu Z, Yu Z (2014b) Discovering and profiling overlapping communities in location-based social networks. IEEE Trans Syst Man Cybern Syst 44(4):499–509
Wang W, Jiao P, He D, Jin D, Pan L, Gabrys B (2016a) Autonomous overlapping community detection in temporal networks. Knowl Based Syst 110(C):121–134

Wang X, Jin D, Cao X, Yang L, Zhang W (2016b) Semantic community identification in large attribute networks. In: Proceedings of the thirtieth conference on artificial intelligence, AAAI'16, pp 265–271

Wasserman S, Faust K (1994) Social network analysis: methods and applications, volume 8 of structural analysis in the social sciences. Cambridge University Press, Cambridge

Whang JJ, Sui X, Dhillon IS (2012) Scalable and memory-efficient clustering of large-scale social networks. In: 2012 IEEE 12th international conference on data mining, ICDM'12, pp 705–714

Whitbeck J, Conan V, Dias de Amorim M (2011) Performance of opportunistic epidemic routing on edge-markovian dynamic graphs. IEEE Trans Commun 59(5):1259–1263

Wu Z, Zou M (2014) An incremental community detection method for social tagging systems using locality-sensitive hashing. Neural Netw 58:14–28

Xie J, Szymanski BK (2013) Labelrank: a stabilized label propagation algorithm for community detection in networks. In: Proceedings of 2013 IEEE 2nd network science workshop, NSW'13, pp 138–143

Xie J, Chen M, Szymanski BK (2013) Labelrankt: incremental community detection in dynamic networks via label propagation. In: Proceedings of the ACM SIGMOD workshop on dynamic networks management and mining, DyNetMM'13, pp 25–32

Yang B, Liu J (2007) An autonomy oriented computing (aoc) approach to distributed network community mining. In: First international conference on self-adaptive and self-organizing systems, SASO'07, pp 151–160

Yang B, Huang J, Liu D, Liu J (2009) A multi-agent based decentralized algorithm for social network community mining. In: 2009 international conference on advances in social network analysis and mining, ASONAM'09, pp 78–82

Yang B, Liu D, Liu J (2010) Discovering communities from social networks: methodologies and applications. In: Furht B (ed) Handbook of social network technologies and applications. Springer, Boston, MA, pp 331–346

Zhang Y, Wang J, Wang Y, Zhou L (2009) Parallel community detection on large networks with propinquity dynamics. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'15, pp 997–1006

Zhao Z, Feng S, Wang Q, Huang JZ, Williams GJ, Fan J (2012) Topic oriented community detection through social objects and link analysis in social networks. Knowl Based Syst 26:164–173

Zhou Y, Cheng H, Yu JX (2009) Graph clustering based on structural/attribute similarities. Proc VLDB Endow 2(1):718–729