Noname manuscript No.
(will be inserted by the editor)

Community Detection in Social Media

Performance and application considerations

Symeon Papadopoulos · Yiannis Kompatsiaris · Athena Vakali · Ploutarchos Spyridonos

Received: date / Accepted: date

Abstract The proposed survey discusses the topic of community detection in the context of Social Media. Community detection constitutes a significant tool for the analysis of complex networks by enabling the study of mesoscopic structures that are often associated with organizational and functional characteristics of the underlying networks. Community detection has proven to be valuable in a series of domains, e.g. biology, social sciences, bibliometrics. However, despite the unprecedented scale, complexity and the dynamic nature of the networks derived from Social Media data, there has only been limited discussion of community detection in this context. More specifically, there is hardly any discussion on the performance characteristics of community detection methods as well as the exploitation of their results in the context of real-world web mining and information retrieval scenarios.

To this end, this survey first frames the concept of community and the problem of community detection in the context of Social Media, and provides a compact classification of existing algorithms based on their methodological principles. The survey places special emphasis on the performance of existing methods in terms of computational complexity and memory requirements. It presents both a theoretical and an experimental comparative discussion of several popular methods. In addition, it discusses the possibility for incremental application of the methods and proposes five strategies for scaling community detection to real-world networks of huge scales. Finally, the survey deals with the interpretation and exploitation of community detection results in the context of intelligent web applications and services.

Keywords community detection · large-scale networks · Social Media

Symeon Papadopoulos · Yiannis Kompatsiaris
Informatics and Telematics Institute, CERTH
E-mail: {papadop,ikom}@iti.gr

Symeon Papadopoulos · Athena Vakali · Ploutarchos Spyridonos
Department of Informatics, AUTH
E-mail: {avakali,plspyrid}@csd.auth.gr

1 Introduction

Networks are omnipresent on the Web. The most profound Web network is the Web itself comprising billions of pages as vertices and their hyperlinks to each other as edges (Kumar et al. [57]). Moreover, collecting and processing the input of Web users (e.g. queries, clicks) results in other forms of networks, such as the query graph (Baeza-Yates [7]). Finally, the widespread use of Social Media applications, such as Delicious[1], Digg[2], Flickr[3] and YouTube[4], is responsible for the creation of even more networks, ranging from folksonomy networks (Mika [72]) to rich media social networks (Lin et al. [66]). Since networks originating from Social Media data are of particular interest to this study, we shall collectively refer to them as *Social Media networks*.

Despite the differences of Social Media networks with respect to the entities and the type of relations they model, they present a significant source of intelligence since they encode the online activities and inputs of masses of Social Media participants. Not only is it possible by analyzing such networks to gain insights into the social phenomena and processes that take place in our world, but one can also extract actionable knowledge that can be beneficial in several information management and retrieval tasks, such as online content navigation and recommendation. However, the analysis of such networks poses serious challenges to data mining methods, since these networks are almost invariably characterized by huge scales and a highly dynamic nature.

A valuable tool in the analysis of large complex networks is *community detection*. The problem that community detection attempts to solve is the identification of groups of vertices that are more densely connected to each other than to the rest of the network. Detecting and analyzing the community structure of networks has led to important findings in a wide range of domains, ranging from biology to social sciences (Girvan and Newman [43]) and the Web (Kumar et al. [56], Flake at al. [34]). Such studies have shown that communities constitute meaningful units of organization and that they provide new insights in the structure and function of the whole network under study. Recently, there has been increasing interest in applying community detection on Social Media networks not only as a means of understanding the underlying phenomena taking place in such systems, but also to exploit its results in a wide range of intelligent services and applications, e.g. automatic event detection in Social Media content.

Despite the increasing significance of Social Media and the proliferation of methods for detecting communities, there has been no prior effort to collect and systematically discuss research efforts with reference to the emerging topic of **community detection in Social Media**. In particular, there are two important aspects of the problem that are not adequately addressed in related survey articles: (a) **performance aspects** of community detection methods, namely computational complexity, memory requirements and possibility for incremental updates of already identified community structure, (b) **interpretation and exploitation** of community detection results by Social Media applications.

Previous related works have studied individual aspects of this problem. For instance, the survey articles by Fortunato [36–38] contain an extensive discussion of numerous community detection methods; however, they are mostly concerned with the

[1] `http://delicious.com`
[2] `http://digg.com`
[3] `http://flickr.com`
[4] `http://www.youtube.com`

methodological foundations of community detection. Furthermore, they describe community detection in a generic context, mostly under the statistical physics perspective, thus lacking any association to web mining and Social Media research. The study by Danon et al. [25] presents a comparative discussion on the computational complexity of several community detection methods. However, it disregards their memory requirements, and other scalability considerations, such as the possibility for incremental computations, and it further lacks a Social Media context. In addition, Schaeffer [94] presents a generic overview of graph clustering, which can be considered as equivalent to community detection. Finally, Tang and Liu [104] discuss several social network analysis problems with emphasis on community detection relating them with the Social Media domain; nevertheless, it is limited to the methodological principles of methods, discussing neither the performance attributes of methods nor the interpretation and exploitation of their results.

The lack of prior work systematically dealing with the aforementioned aspects of community detection in relation to Social Media applications has motivated the present study, which makes the following contributions:

- Frame the **concept of community** and the problem of community detection in the context of **Social Media** (Section 3).
- Present a systematic study and a **compact methodological classification** of several community detection methods (Section 4.1).
- Study existing methods in view of the scalability challenges posed by the magnitude and dynamic nature of Social Media networks. Compare **complexity and memory requirements** of existing methods (subsection 4.2) as well as **incremental/dynamic computation** characteristics (subsection 4.3).
- Conduct a comparative experimental study (subsection 4.2.2) benchmarking eight popular community detection algorithms in terms of execution time, peak memory usage and attained community structure precision.
- Introduce **five scalability strategies** that can be employed in order to scale community detection methods to the magnitude of Social Media data (Section 5).
- Discuss the **interpretation and exploitation of community detection results** by Social Media applications and services (Section 6).

Due to the breadth of the topic under study, we do not attempt to be exhaustive with respect to the number of presented methods, nor do we target at delving into deep foundational issues of the methods. Instead, we aim at discussing the methodological, performance and application aspects of existing techniques under a paradigmatic and comparative perspective, which we believe will be valuable for applying them in Social Media mining problems.

2 Background

The term "community" has been extensively used in the literature in different contexts and with different connotations. Social studies are probably the earliest context, where the notion of community was used to denote groups of people with shared interests or activities (e.g. *communities of practice*). Once networks became widely adopted as a means to study social interactions and processes (Wasserman and Faust [109], Scott [97]), the concept of community was associated to networks of human actors exhibiting certain characteristic structural properties.

As networks became established as a model for many real-world complex systems, the concept of community expanded in scope to refer to group structures in a variety of networks, not necessarily consisting of human actors. For instance, community structure was studied in the context of protein interaction networks, bibliographical citation networks, networks of football teams, and others. With the emergence of the Web and the Web Graph as a prevalent model of web pages and their hyperlinks, communities were also considered on the Web (Kumar et al. [56], Flake et al. [34]).

The advent and wide success of Social Media has created a new context for the notion of community. Currently, there is a variety of online entities residing in the virtual setting of Social Web applications and there are numerous kinds of interactions and relations among such entities. For that reason, before proceeding with the definition of communities in Social Media in Section 3, we will first describe the concept of Social Media networks and the elements that they comprise (subsection 2.1), as well as the process of their creation (subsection 2.2).

2.1 Elements of a Social Media network

The ecosystem of Social Media applications comprises a wide range of objects that are associated to each other through numerous types of interactions and relations. Social Media networks provide an elegant representation of Social Media data, containing online objects as their vertices and the relations/interactions among them as edges[5]. The vertices of Social Media networks can represent different types of actors, such as users, content items (e.g. blog posts, photos, videos), and even metadata items (e.g. topic categories, tags). In addition, the edges of Social Media networks can be of different types, such as simple, weighted, directed and multiway (i.e. connecting more than two entities) depending on the network creation process (discussed below).

In terms of notation, Social Media networks employ the typical graph notation $G = (V, E)$, where G stands for the whole network, V stands for the set of all vertices and E for the the set of all edges. Due to the different types of vertices and edges in such networks, it is common to consider sets of vertices and edges within V and E that contain vertices and edges of the same type. For instance, in the case of a photo sharing and tagging network, one can consider the set of vertices V to comprise the users, photos and tags of the system, i.e. $V = \{U, P, T\}$. Similarly, the set of edges in such an application would comprise the set of user-photo, photo-tag and user-tag associations, $E = \{UP, PT, UT\}$.

2.2 Social Media network creation

In practice, the creation of Social Media networks starts from a set of transactions that are performed and recorded in Social Media applications. Every such transaction typically involves different entities; for instance a *tag assignment* in Flickr involves a user, a photo and a tag, while a *comment* on a blog article involves the commenter, the blog article and the comment text. In that way, an association (edge) is formed between the items of the same transaction on an underlying network, so that the resulting Social Media network constitutes a direct representation of a subset of online transactions.

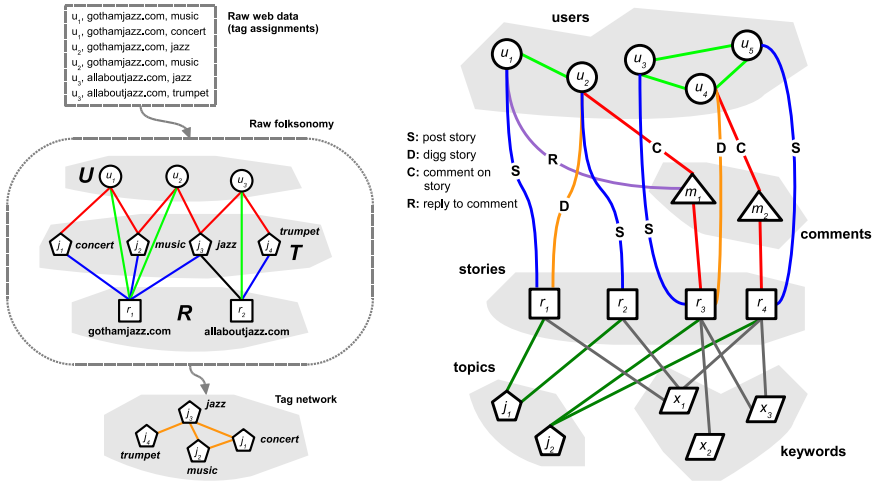
[5] There are alternative data models, such as multidimensional data cubes (Chi et al. [22]), for representing Social Media data. However, those models are not considered in this survey.

Since "raw" Social Media networks comprise multiple types of vertices and edges (some of which can be multi-way, i.e. link more than two vertices), they are mathematically represented by hypergraphs. Alternatively, if each hyper-edge of the graph is reduced to the pairwise connections among the k different types of nodes, the hypergraph is reduced to a k-partite graph. The majority of network analysis methods, and community detection in particular, are not applicable to hypergraphs or k-partite graphs. For that reason, it is common practice to extract simplified network forms that depict partial aspects of the complex interactions of the original network. Such networks are typically one- or two-mode (i.e. contain only one or two vertex types) and contain simple edges (i.e. connecting two vertices), which makes possible the application of numerous network analysis techniques. In summary, the typical lifecycle of a Social Media network (Figure 1) involves its creation from a set of recorded transactions and its transformation into some suitable form for the analysis that follows.

Folksonomies constitute an extensively studied example of Social Media networks. A folksonomy comprises three types of entities, namely users, resources and tags (Mika [72]). Starting from the tag assignments of users, i.e. transactions involving a user, a resource and a tag, a "raw" folksonomy network is formed, comprising three types of vertices and three-way relations among them. Subsequently, simpler (two-mode or one-mode) network representations are derived by use of projection operations (Mika [72], Schmitz et al. [96]). For instance, Figure 2(a) presents a toy folksonomy network created from six tag assignment transactions. The "raw" tri-partite folksonomy network is transformed to a simple tag association network by considering an edge between two tags when they are used to tag the same resource. Other variants of deriving tag association networks from folksonomies are described in the works by Cattuto et al. [16], and Au Yeung et al. [6].

A more sophisticated paradigm for Social Media networks, termed *metagraph*, is presented by Lin et al. [66]. The vertices of a metagraph are organized in *facets* and interactions among different facets, which can be multi-way, constitute the edges of the metagraph. For instance, a metagraph representation of the Digg Social Media application considers users, stories, comments, topics and keywords as different facets of the metagraph representation and further maps user activities, such as votes and commenting, to edges on the metagraph as illustrated in Figure 2(b). Another network model is proposed by Agichtein et al. [1] for representing users, questions and answers in a community question-answering application. In conclusion, different forms of Social Media networks are possible depending on the transactions of the Social Media application under study, the modeling requirements of the problem, as well as the capabilities of the network analysis method at hand.

Fig. 1 Typical lifecycle of a Social Media network: A set of transactions involving users, content and metadata lead to the formation of a "raw" Social Media network. Typically, this network is simplified before any sophisticated analysis (e.g. community detection) takes place.



(a) Folksonomy (b) Digg metagraph

Fig. 2 Two paradigmatic Social Media networks: (a) a folksonomy network (Mika [72,96]) (b) a digg metagraph (Lin et al. [66]).

3 Social Media communities

Due to the abundance of related works and the variety of adopted perspectives, there is no unique and widely accepted definition of community. Community definitions are formulated with reference to the network structure of the system under study and are commonly bound to some property either of some set of vertices (local definitions) or of the whole network (global definitions). However, at a different level, one should also define a community with respect to the domain under study, which in this survey comprises the realm of Social Media systems. For that reason, we will first provide a qualitative definition of Social Media communities and subsequently we will link this definition to established network-based definitions of quantitative nature.

At the most abstract level, given a Social Media network $G = (V, E)$, a **Social Media community** can be defined as a subgraph of the network comprising a set $V_C \subseteq V$ of Social Media entities that are associated with a common element of interest. This element can be as varied as a topic, a real-world person, a place, an event, an activity or a cause. For instance, in a blogging network, the set of all bloggers, articles, tags and comments related to the topic of "renewable energy" constitutes the respective community. Similarly, in a photo sharing application, the set of users, photos and tags that are associated with the island of Crete form a distinct community.

Social Media communities can further be described as *explicit* or *implicit*. Explicit communities are created as a result of human decision and acquire members based on human consent. Examples of explicit Social Media communities are Facebook and Flickr Groups. Implicit communities, on the other hand, are assumed to exist in the system and "wait" to be discovered. Implicit communities are particularly important for two reasons: (a) they do not require human effort and attention for their creation and (b) they enable the study of emerging phenomena within Social Media systems. This survey focuses on the definition and discovery of implicit communities.

3.1 Classes of implicitly defined communities

Implicit communities are defined with reference to the network structure. The most established notion of community-ness within a network is based on the principle that some sets of vertices are more densely connected to each other than to the rest of the network. Depending on whether this property of vertices is considered locally (on a connected subset of vertices) or globally (on the whole network), we distinguish between *local* and *global* community definitions. Communities are also defined on the basis of the result of some principled network-based process (*process-based* definitions). Several of these definitions are tabulated and described in the supplementary material included in the work by Kovács et al. [55].

Local definitions. Early notions of community emerged from social studies focusing on the concepts of subgroup cohesiveness and mutuality. Examples of such community definitions are *cliques*, *n-cliques*, *n-clubs*, *n-clans*, *k-plexes*, and *k-cores* (Wasserman and Faust [109], Scott [97]). More recently, the structures of LS and Lambda sets were defined (Borgatti et al. [13]) as community constructs. However, most of the above definitions are too restrictive and computationally very expensive. Thus, their use is very limited in a Social Media context.

Alternatively, the internal and external vertex and subgraph degrees have been used to define community-ness. The internal degree of a vertex is the number of edges that connect it to vertices of the same community. The external degree is defined in a similar way. The definitions of communities in the *strong* and *weak* sense (Radicchi et al. [88]) are based on the internal and external degrees of vertices belonging to a community. Moreover, it is possible to define some local measure of community quality and then quantify the degree of community-ness for a given subgraph. Examples of such local measures are local and relative density (Šíma et al. [100]), local modularity (Clauset [24]), and subgraph modularity (Luo et al. [68]).

Global definitions. Global community definitions consider community structure as a property of the whole network. There are several important classes of global community definitions. Perhaps, the most intuitive community definition relies on the number of edges falling between communities (cut size) as a measure of quality of a given network partition into communities. Since the absolute number of inter-community edges is problematic, normalized measures such as *Normalized Cut* (Shi and Malik [99]) and *conductance* (Kannan et al. [52]) have been introduced for quantifying the profoundness of separation between the communities of a network.

Another class of global community structure is based on the widely used concept of *modularity* that quantifies the extent to which a given partition of a network into communities deviates from the hypothetical state (null model) that the network would be randomly rewired under the constraint of same-degree for each vertex (Newman and Girvan [74]). A generalization of modularity was presented by Reichardt and Bornholdt [90] and numerous variations of the concept have been introduced for computing modularity in weighted (Newman [76]), directed (Arenas et al. [4]) and bipartite (Barber [9]) networks.

Finally, a wide class of global community definitions relies on some similarity measure between network vertices. Once pairwise similarities between vertices are computed, communities are defined as clusters of vertices that are close to each other. Vertex similarities can be derived by use of numerous methods, such as embedding

graph vertices in n-dimensional Euclidean space (and then use some traditional distance measure such as Manhattan or Euclidean distance) or using the adjacency matrix of the graph, e.g. compute Pearson correlation between rows of the adjacency matrix, or random-walk based similarities (Pons and Latapy [85]). Another vertex similarity measure is the *structural equivalence* (Lorrain and White [67]), which expresses the overlap between the neighborhoods of two vertices (even if they are not connected).

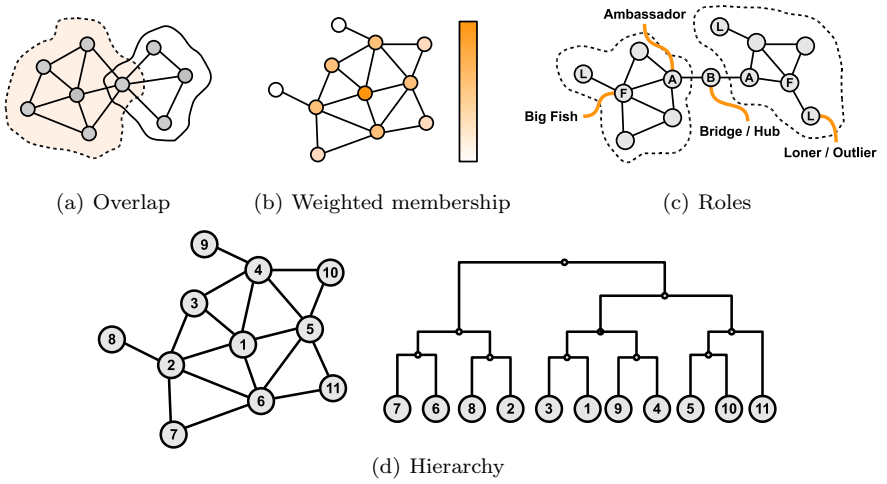
Process-based definitions. An alternative means of defining communities is by considering some community formation process taking place on the network under study. For instance, the Clique Percolation Method (Palla et al. [78]) considers a k-clique template that "rolls" on the network and results in a community consisting of the union of all k-cliques that are adjacent to each other (i.e. share $(k-1)$ nodes). Other community definitions rely on a dynamic process that is iteratively applied on the network in order to reveal groups of vertices that form well-separated communities. Van Dongen [30] describes a flow diffusion process, namely the Markov Cluster Algorithm, which is iteratively applied on the network in order to render the underlying community structure conspicuous. According to it, communities are defined as sets of vertices, in which a random walker is very likely to get trapped. Another dynamic process used for the definition of communities is the synchronization of a set of phase oscillators on a network (Arenas et al. [3]): groups of vertices whose oscillators synchronize first are considered to form communities. Finally, a label propagation scheme based on neighbor majority voting is devised by Raghavan et al. [89] to define communities as groups of vertices forming stable consensus with respect to their label.

3.2 Community attributes

Across all community definitions presented above, a set-based view of communities was presented: community was seen as a set of vertices and the membership of each vertex in a network was implicitly assumed to be the result of a boolean decision. In reality, and especially in the context of Social Media, the concept of community and community membership may be more complicated. For instance, in several of the above community definitions, e.g., most of the local ones (Clauset [24], Luo et al.[68]), the one based on Clique Percolation (Palla et al. [78]), and others (Gregory [45], Chen et al. [21]), it is possible for communities to overlap (Figure 3(a)). Community overlap is important for Social Media networks since it is common for Social Media entities to participate in multiple communities; for instance, a user may be affiliated to his/her family, friends and professional community.

In addition, there are other attributes that vertices of a network may have in relation to communities. For instance, different vertices may participate with varying degrees in a community depending on their centrality[6] within it (Figure 3(b)). Moreover, vertices may have discrete roles: for example, Xu et al. [110] define two roles (*hubs* and *outliers*) for vertices that are not assigned to any community. Hubs are connected to multiple communities and act as liaisons, thus enabling interactions among communities. Outliers are connected to a single community through a single link, therefore they are usually considered as noise. Community-based vertex roles are also discussed by Scripps et al. [98]. Specifically, the roles roles of "loners", "big fish", "bridges" and "ambassadors" are defined (Figure 3(c)).

[6] Centrality quantifies how frequently vertices fall in the paths between other vertices.



(a) Overlap (b) Weighted membership (c) Roles

(d) Hierarchy

Fig. 3 Several attributes that may characterize community structure: (a) overlap, (b) weighted membership, (c) vertex roles within/across communities, (d) hierarchical organization.

Finally, it is possible to impose hierarchical (Figure 3(d)) or multi-scale structures on communities. Community organization may be considered at different scales in a variety of systems. For instance, a set of users of a Social Media application may be organized in a community focused on a very specific topic (e.g. fans of a particular indie-rock band) and at the same time they may be considered as members of a broader community (rock music). For Social Media systems, the consideration of multiple levels of community organization typically does not involve any kind of hierarchical organization since the constraints imposed by the hierarchical model are too restrictive for modeling the uncontrolled and emerging nature of Social Media phenomena.

4 Community detection methods

From the discussion of Section 3 it became clear that there is a variety of community definitions based on network measures and structures. The variety of methods that have appeared in literature for detecting communities is even larger, since for each community definition there are more than one methods claiming to detect the respective communities. Here, we will summarize the most important classes of such methods, associate them with the definitions presented in the previous section and comparatively discuss their performance requirements in terms of computational complexity and memory consumption, as well as their dynamic computation characteristics, which are particularly pertinent for the analysis of Social Media networks. Before proceeding with the discussion of the methods, we will first delineate the relation of the problem of community detection with that of graph partitioning and graph clustering.

Community detection and graph partitioning. Graph partitioning is a well-specified problem: divide the vertices of a graph into n groups of given sizes such that the number of edges lying between the groups (cut size) is minimum. Community detection is different from graph partitioning in two fundamental aspects. First, community detection requires neither the number of groups nor their sizes as input in

order to extract them. Furthermore, the result of community detection may not be a partition, i.e. a set of vertex sets, whose union is the set of all graph vertices and whose pairwise intersections result always in the empty set. As became apparent in Section 3, communities in a network may present overlap and there may be vertices in a network that are not assigned to any community.

Community detection and graph clustering. Community detection is almost interchangeably used with graph clustering (Fortunato [38], Schaeffer [94]). In both problems, the aim is to identify groups of vertices on a graph that are better connected to each other than to the rest of the network. However, a differentiation between the two problems regards the requirement for knowing the number of communities/clusters that a method is expected to identify. Community detection methods typically do not require the number of communities to be provided as input, but instead the number of communities is one of the method outputs[7]. In contrast, there are numerous graph clustering techniques that require the number of clusters to be provided as input. Due to the immense scale and evolving nature of Social Media, it is almost impossible to know or even to estimate the number of communities in a Social Media network. Therefore, graph clustering methods that require the number of clusters to be provided as input are of limited use for the study of such data[8].

In this section, we first present a classification of existing community detection and graph clustering methods based on their methodological principles (subsection 4.1). Furthermore, we provide a discussion and experimental study of their computational complexity and memory requirements (subsection 4.2), which are particularly pertinent for their application on a Social Media scale. Finally, we discuss the potential of applying existing community detection methods on dynamic networks (subsection 4.3).

4.1 Methodological classification

Depending on the underlying methodological principle as well as the adopted definition of community, we consider five broad classes of community detection and graph clustering methods: (a) cohesive subgraph discovery, (b) vertex clustering, (c) community quality optimization, (d) divisive, and (e) model-based. For the sake of self-containment, we are going to briefly refer to each of the aforementioned method classes here. For a more thorough discussion of their principles, we refer to the survey articles by Danon et al. [25], Fortunato [36–38], Porter et al. [86], and Schaeffer [94]. Also, a useful listing of a large number of community detection methods appears in the supplementary material of Kovács et al. [55].

Cohesive subgraph discovery. The methods of this class presume a specification of the structural properties that a subgraph of the network should satisfy in order to be considered a community. Once such a subgraph structure is specified, methods involve the enumeration of such structures in the network under study. The local community definitions presented in subsection 3.1, e.g., cliques, n-cliques, k-cores, LS sets and lambda sets, are examples of such cohesive structures and therefore algorithmic schemes for enumerating such structures, such as the Bron-Kerbosch algorithm (Bron

[7] That does not imply that community detection methods are necessarily parameter-free. They may well rely on some other parameter.

[8] However, in case it is possible to identify a relatively small set of values for the number of clusters to be detected, it is customary to employ such algorithms within some iterative parameter selection scheme with the goal of identifying the optimal number of clusters.

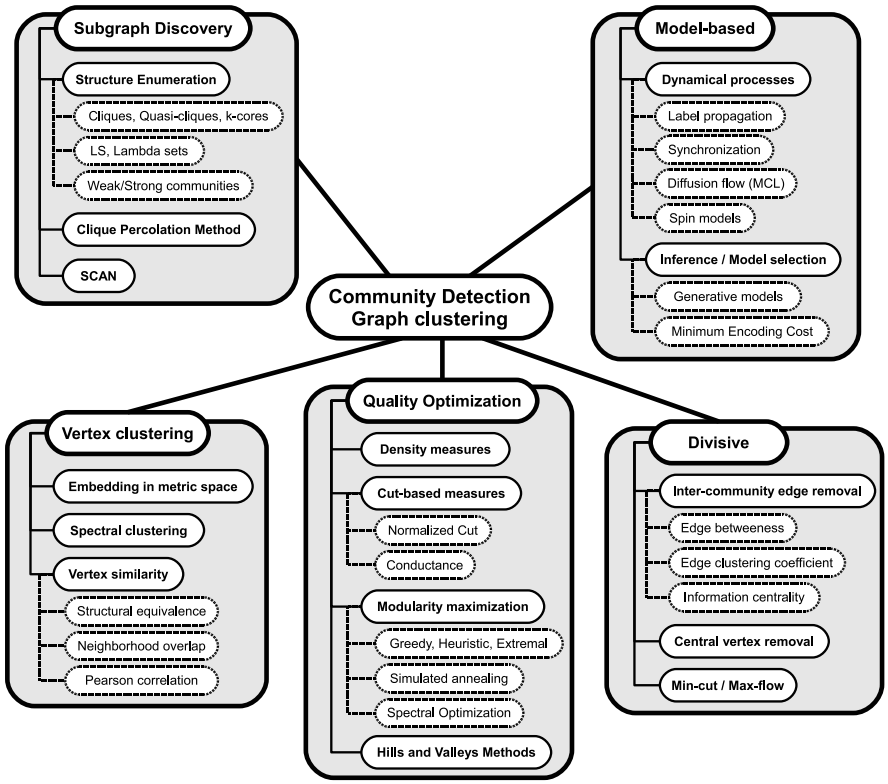


Fig. 4 A classification of community detection and graph clustering methods according to the adopted community definition and underlying methodological principle.

and Kerbosch [15]) and the efficient k-core decomposition algorithm of Batagelj and Zaversnik [10], belong to this class of community detection methods. In addition, methods such as the Clique Percolation Method (Palla et al. [78]) and the SCAN algorithm (Xu et al. [110]), which lead to the discovery of subgraph structures with well-specified properties, fall under the same class of methods.

Vertex clustering. Such techniques originate from the traditional data clustering research. A typical means of casting a graph vertex clustering problem to one that can be solved by conventional data clustering methods (such as k-means and hierarchical agglomerative clustering) is by embedding graph vertices in a vector space, where pairwise distances between vertices can be calculated. Another popular method is to use the spectrum of the graph for mapping graph vertices to points in a low-dimensional space, where the cluster structure is more profound (Donetti and Munoz [29], Von Luxburg [69]). Other vertex similarity measures such as the structural equivalence (Breiger et al. [14]) and the neighborhood overlap have been used to compute similarities between graph vertices (Wasserman and Faust [109]). Finally, a noteworthy method, called Walktrap (Pons and Latapy [85]), makes use of a random-walk based similarity between vertices and between communities and uses modularity in a hierarchical agglomerative clustering scheme to derive an optimal vertex clustering structure.

Community quality optimization. There is a very large number of methods that are founded on the basis of optimizing some graph-based measure of community quality. Subgraph density and cut-based measures, such as normalized cut (Shi and Malik [99]) and conductance (Kannan et al. [52]), were among the first to be used for quantifying the quality of some network division into clusters. A whole new wave of research was stimulated by the measure of modularity. Approximate modularity maximization schemes abound in the literature. Apart from the seminal greedy optimization technique of Newman [75], and speeded up versions of it, such as max-heap based agglomeration (Clauset et al. [23]) and iterative heuristic schemes (Blondel et al. [12]), more sophisticated optimization methods have been devised, for instance, extremal optimization (Duch et al. [31]), simulated annealing (Massen and Doye [71]) and spectral optimization (Newman [77]). Methods aiming at the optimization of local measures of community quality, such as local and subgraph modularity (Clauset [24], Luo et al. [68]), also belong to this category. Finally, this category includes methods that exploit the "hills" and "valleys" in the distribution of network-based node or edge functions, e.g. the ModuLand framework proposed by Kovács et al. [55] and the "reachability" measure by Chen et al. [21].

Divisive. These methods rely on the identification of network elements (edges and vertices) that are positioned between communities. For instance, the seminal algorithm by Girvan and Newman [43] progressively removes the edges of a network based on some edge betweeness measure until communities emerge as disconnected components of the graph. Several measures of edge betweeness have been devised, for instance, edge, random-walk, and current-flow betweeness (Newman and Girvan [74]), as well as information centrality (Fortunato et al. [35]) and the edge clustering coefficient Radicchi et al. [88]). A similar principle is adopted by vertex removal methods (Vragović and Louis [107]); such methods remove vertices in order to reveal community structure. Finally, min-cut/max-flow methods (Flake et al. [34], Ino et al. [49]) adopt a different divisive perspective: they try to identify graph cuts (i.e. sets of edges that separate the graph in pieces) that have a minimum size.

Model-based. This is a broad category of methods that either consider a dynamic process taking place on the network, which reveals its communities, or they consider an underlying model of statistical nature that can generate the division of the network into communities. Examples of dynamic processes are label propagation (Raghavan et al. [89], Leung et al. [62], Gregory [45]), synchronization of Kuramoto oscillators (Arenas et al. [3]), diffusion flow, better known as Markov Cluster Algorithm (Van Dongen [30]), and the popular spin model by Reichardt and Bornholdt [90]. In addition, community detection can be cast as a statistical inference problem (Hastings [46]), assuming some underlying probabilistic model, such as the planted partition model, that generates the community structure and estimating the parameters of this model. Other model-based approaches rely on the principle that a good clustering is determined by a low encoding cost, thus they perform community detection by finding the cluster structure that results in the lowest possible cluster encoding cost (Chakrabarti [18], Rosvall and Bergstrom [92]).

4.2 Performance comparison

In assessing the performance of community detection methods, there are two fundamental aspects that one needs to consider: (a) the computational complexity and (b)

the requirements of the method in terms of main memory. As will be discussed in the following (subsection 4.3), the incremental computation properties of community detection methods constitute an additional performance consideration, which is increasingly important in the context of Social Media systems. In this section, we provide a theoretical discussion of the first two performance aspects, computational complexity and memory requirements, for a variety of methods, and report the results of an experimental study that compares the performance of eight popular community detection methods on a wide variety of synthetic networks.

4.2.1 Theoretical discussion

Computational complexity: It is customary to quantify the computational complexity of community detection and graph clustering algorithms in terms of graph parameters. Since many of the algorithms rely on vertex and/or edge iteration schemes, their complexity is commonly expressed in relation to the number of graph vertices n and/or edges m. In addition, the complexity of algorithms is sometimes dependent on the average or maximum degree of the graph vertices, denoted by $\overline{k}$ and k_{max} respectively. Sometimes, the complexity also depends on the number C of communities to be discovered. For hierarchical methods that produce a *community dendrogram*, the depth d of the dendrogram can also affect the complexity of the method.

In the literature, there are many cases where, instead of the "true" complexity of an algorithm, one reports its approximate complexity based on the assumption that the underlying graphs are sparse, i.e. $n \approx m$. Since this distinction is frequently not evident, we are going to present, for each method, the complexity both in the case where no assumption is made about the underlying graph and in the case where the assumption of a sparse graph is made. Table 1 presents a complexity comparison among several selected graph clustering and community detection methods. From the table, it appears that vertex clustering and divisive approaches present complexities higher than quadratic to the number of network vertices, which renders them inapplicable for large- and mega-scale networks that commonly appear in Social Media systems. On the other hand, there are several recent methods (Blondel et al.[12], Leung et al. [62], Raghavan et al. [89], Xu et al. [110]) from the other classes that present linear complexity to the number of network edges. Since the number of edges contained in real-world Social Media networks can easily exceed the billion scale, it is obvious that such methods are preferable in terms of complexity for tackling community detection on such networks.

For several popular methods, it is not possible to derive a closed-form expression of their complexity, as this depends on the structure of the graph under study, as well as on particular method parameters. For instance, in the case of the Clique Percolation Method (Palla et al. [78]) there is no closed-form expression for bounding the algorithm running time, although it is experimentally demonstrated that the algorithm completes in reasonable time for fairly large networks. In a similar fashion, the complexity of the modularity maximization method based on simulated annealing (Massen and Doye [71]) cannot be expressed in terms of graph characteristics (although it is known to be a slow method and applicable only to relatively small networks).

A whole class of methods whose complexity cannot be expressed in closed form are the methods that rely on spectral graph properties (Von Luxburg [69]). Such methods are based on the computation of some or all the eigenvectors of the graph Laplacian in order to cluster the vertices of the graph. In the worst case, where all eigenvectors are computed, such methods have a $O(n^3)$ complexity. However, most methods compute

Table 1 Comparison of community detection complexity. Two bounds are provided, one for general graphs irrespective of density (Complexity-A) and one computed under the assumption that the graph is sparse (Complexity-B). Furthermore, the scale of graphs for which each method is appropriate is provided: S stands for small scale ($< 10^4$ nodes), M stands for medium scale ($< 10^6$ nodes), and L for large scale ($10^6 - 10^9$ nodes).

Method	**Complex-A**	**Complex-B**	**Scale**
Cohesive substructure detection			
Bron-Kerbosch [15]	$O(3^{n/3})$	$O(3^{n/3})$	S
k-core detection (Batagelj and Zaversnik [10])	$O(n^2)$	$O(n)$	L
SCAN (Xu et al. [110])	$O(n^2)$	$O(n)$	L
Vertex clustering			
embedding in space + k-means	$O(C\,n^2)$	$O(C\,n^2)$	M
Walktrap (Pons and Latapy [85])	$O(n^4)$	$O(n^2 \log n)$	M
Donetti-Munoz [29]	$O(C\,n^2)$	$O(C\,n^2)$	M
Community quality optimization			
Clauset-Newman-Moore [23]	$O(n^2\,d\,\log n)$	$O(n\,\log^2 n)$	M
Extremal optimization (Arenas et al. [4])	$O(n^2 \log n)$	$O(n^2 \log n)$	M
Spectral optimization (Newman [77])	$O(n^3\,d)$	$O(n^2 \log n)$	M
Community folding (Blondel et al. [12])	$O(n^2)$	$O(n)$	L
Divisive			
Girvan-Newman [43]	$O(n^5)$	$O(n^3)$	S
Information centrality (Fortunato et al. [35])	$O(n^7)$	$O(n^4)$	S
Edge clustering coefficient (Radicchi et al. [88])	$O(n^6)$	$O(n^2)$	M
Max flow + Gomory-Hu tree (Ino et al. [49])	$O(n^4\,\log n)$	$O(n^3 \log n)$	S
Model-based			
MCL (Van Dongen [30])	$O(n^3)$	$O(n^3)$	M
Minimum encoding cost (Chakrabarti [18])	$O(n^2)$	$O(n)$	L
Label propagation (Raghavan [89], Leung [62])	$O(n^2)$	$O(n)$	L
Infomap (Rosvall and Bergstrom [92])	$O(n^2 \log n)$	$O(n \log n)$	L

only a limited number of eigenvectors (corresponding to the smallest eigenvalues of the Laplacian) by use of some efficient method (e.g. Lanczos) and are thus much more efficient. Still, their complexity cannot be expressed in relation to the graph size.

Finally, the complexities of methods that operate at a local level, such as the ones by Clauset [24], Luo et al. [68], Bagrow [8], Papadopoulos et al. [80], were not included in Table 1, since such methods are used to discover a single community around a given vertex or set of vertices. Although it is possible in principle to derive global community detection schemes by bootstrapping such methods, there is no meaning in comparing the complexity of such schemes with inherently global approaches. For that reason, and due to the fact that local methods can be used as a means of scaling community detection to larger networks, we present an additional discussion of local methods in subsection 5.2.

Memory requirements: The magnitude of networks formed from Social Media data, and the development of community detection methods presenting linear complexity with respect to the network size, has revealed an additional bottleneck to community detection methods: memory consumption. Most existing techniques consider direct and

instant access to graph elements, since the whole graph is assumed to reside in main memory. Moreover, additional main memory structures are frequently used by methods to speed up certain graph analysis operations. For that reason, it is necessary to assess community detection methods on the basis of their memory requirements.

A large number of community detection methods rely on the edges of the graph under study to reside in memory. In addition, the assignments of nodes to communities and their degrees need to be readily accessible by the algorithm (e.g. in the case of modularity computation). Thus, the bare minimum memory consumption for many community detection methods scales linearly with the size of the graph[9] $(2 \cdot m + 2 \cdot n)$. This estimate assumes that the graph is unweighted and stored in the form of an edge list (thus requiring only two elements per edge, the endpoint vertices). In practice, many community detection implementations rely on a matrix representation of the graph (adjacency matrix) to be available in memory, thus calling for at least $2 \cdot m + 3 \cdot n$ in case of unweighted undirected graphs in case the Yale sparse matrix format is used.

More often than not, methods require additional data structures to be held in memory in order to speed up certain operations. For instance, the efficient greedy scheme of Clauset et al. [23] makes use of a ΔQ matrix instead of a simple adjacency matrix that requires both n binary trees and n max-heaps (which are also represented in the form of trees), thus raising the memory needs of the method. The eigenvector-based approach for modularity maximization (Newman [77]) poses even higher requirements for memory, since it relies on the so-called modularity matrix B that is not sparse and thus takes up n^2 space. Other spectral approaches (Von Luxburg [69]) store a set C of eigenvectors in main memory (in addition to some graph-derived sparse matrix, such as the Laplacian) resulting in the need for additional $C \cdot n$ memory space. Some recent methods tackle the problem of minimizing the Normalized Cut of a clustering (spectral partitioning), without the need to compute the eigenvectors of graph-derived matrices (Dhillon et al. [27]).

Another class of methods that inherently suffer from large memory requirements are those based on vertex clustering. Such methods involve the embedding of graph vertices in some n-dimensional space, thus resulting in a dense $n \times n$ matrix, which takes up n^2 memory space. Moreover, community detection based on statistical inference (Hastings [46]) also suffers from high memory usage. Approximating the solution to the belief propagation problem that the method tries to solve requires an additional $C \cdot n$ memory in order to track the estimated probabilities with which network vertices belong to each community.

On the other hand, there are numerous community detection methods that are suitable even under strict memory constraints. For instance, the label propagation method of Raghavan et al. [89] proceeds by inspecting only the neighborhood of a vertex each time a relabeling operation is performed. Therefore, if needed the method can work even with as little as $n + k_{max}$ memory space (community assignment vector plus vector of largest vertex neighborhood) assuming that the graph is streamed in memory in the form of vertex adjacency lists. Obviously, local community detection approaches are also memory efficient, since they process only a portion of the graph at a time. However, when the local community boundary grows large (e.g. in the case of gigantic communities), even local methods might cause significant memory overhead (this is further discussed in subsection 5.2).

[9] When discussing memory consumption, it is meaningless to use the Big O notation, since we want to distinguish between a memory consumption of $\alpha \cdot m$, $\alpha \in \mathbb{N}_+$ and one of m.

4.2.2 Experimental study

In order to gain insight into the performance aspects of community detection, we have conducted a comparative experimental study including eight well-known methods, namely:[10] (1) the greedy modularity maximization scheme by Clauset, Newman, Moore (CNM) [23], (2) the Walktrap method by Pons and Latapy (WALKTRAP) [85], (3) the "leading eigenvector" method by Newman (LDEIGEN) [77], (4) the spin configuration method by Reichardt and Bornholdt (SPIN) [90], (5) the label propagation method by Raghavan et al. (LPROP) [89], (6) the Markov Cluster Algorithm by van Dongen (MCL) [30], (7) the heuristic modularity maximization scheme by Blondel et al. (LOUVAIN) [12], and (8) the information theoretic method by Rosvall and Bergstrom (INFOMAP) [92].

In order to test the method performance under a wide range of structural network features, we used the synthetic benchmark graphs by Lanchicinetti et al. [58]. The graph generation process is based on five parameters: (a) number of nodes, N, (b) average degree of nodes, $\hat{k}$[11], (c) mixing parameter, μ, indicating the "conspicuousness" of communities (the higher its value, the fuzzier the communities), (d) power law exponent for node degree distribution, γ, (e) power law exponent for community size distribution, β. For testing the effect of each parameter on the performance of the methods, we created synthetic graphs with varying values for each one of the parameters while keeping the rest of the parameters fixed. For each parameter set, we created 10 graph realizations in order to obtain more reliable performance measurements by averaging over the respective measurements. Our performance measurements include execution time, Normalized Mutual Information (NMI)[12], and peak memory consumption for each method. Thus, apart from measuring the computational requirements of the methods, we also benchmarked by means of NMI the quality of the community structure that they produce.

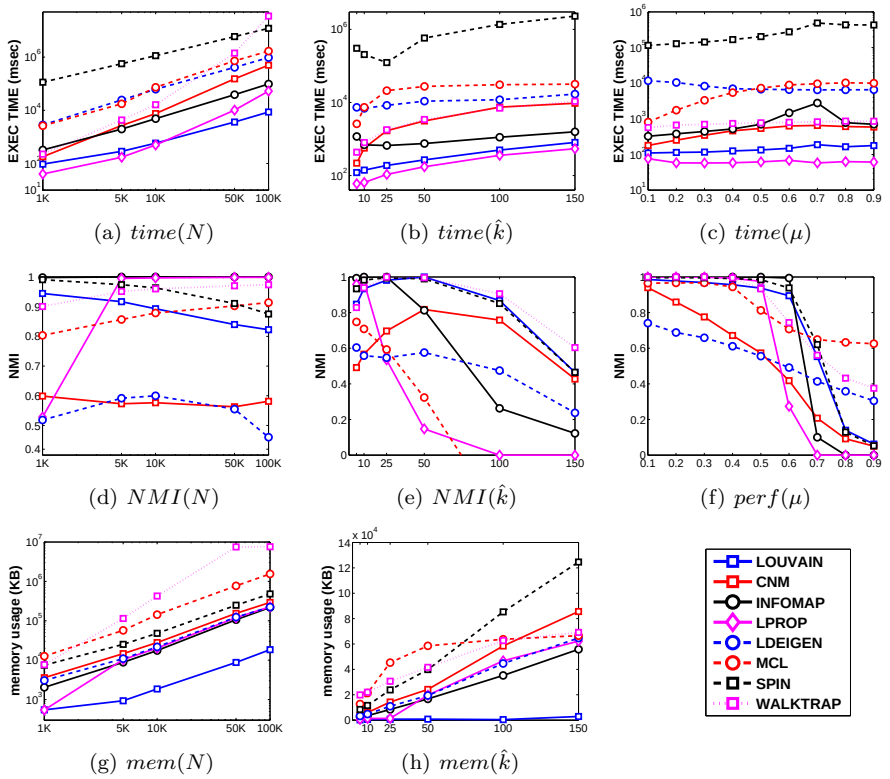
Figure 5 provides an overview of the benchmark results by depicting the relation of method performance to three graph generation parameters, namely N, $\hat{k}$, and μ. For the sake of brevity and due to the fact that there was no significant influence of performance on parameters γ and β, the respective diagrams were omitted. Figures 5(a)- 5(c) depict the relation of the method execution time in relation to the three graph generation parameters. Figures 5(d)- 5(f) illustrate the execution time in relation to the achieved community structure quality in terms of NMI. Finally, Figures 5(g) and 5(h) present the peak memory requirements of the competing methods in relation to the number of nodes and average degree (i.e. number of edges) of the network.

In terms of execution time, there is a remarkable performance difference between the fastest methods, namely LPROP and LOUVAIN, and the slowest ones, i.e. SPIN and MCL. For instance, for a network of 10 thousands of nodes, LPROP and LOUVAIN take approximately 0.5 and 0.6 seconds to complete, while MCL takes 75 seconds and SPIN almost 19 minutes. In terms of scalability, all methods appear to scale almost linearly to the number of nodes in the network. An exception is the WALKTRAP method that exhibits an abrupt increase in the execution time after the network grows larger than 10 thousands of nodes. This is actually not an inherent complexity characteristic of

[10] For ease of reference, we provide for each method a short name in parentheses.

[11] This is equivalent to the number of edges of a network.

[12] This measure is typically used in community detection studies, e.g. the comparative study by Danon et al. [25], to quantify the extent to which the detected community structure matches the "true" one, which in the case of synthetic graphs is known.



(a) $time(N)$ (b) $time(\hat{k})$ (c) $time(\mu)$

(d) $NMI(N)$ (e) $NMI(\hat{k})$ (f) $perf(\mu)$

(g) $mem(N)$ (h) $mem(\hat{k})$

Fig. 5 Benchmark results for eight community detection methods.

the method but it is due to the fact that the method consumes all available memory, as becomes obvious from Figure 5(g), and it thus needs to frequently access the disk. Another method that scales a little worse than linear is the LPROP method, which for small size networks is the fastest one, but for larger size networks comes second after the LOUVAIN method. The execution time of some methods is also affected by the average degree of the network. SPIN, MCL, CNM and WALKTRAP are the methods affected most by the average network degree, while INFOMAP, LOUVAIN and LPROP do not seem to be significantly influenced. Finally, changing the network mixing parameter, i.e. how much the different communities "blend" with each other, does not seem to have significant influence on the execution time of methods with the exception of INFOMAP, MCL and SPIN.

In terms of memory consumption, WALKTRAP was found to be the most memory hungry method, consuming all available 8GB of the test machine when networks of 50 thousands of nodes or larger were analyzed. This was also the reason for the steep decrease in its performance in terms of execution time, as described above. MCL was the second most memory hungry method needing more than 1.5GB of memory for a network of 100 thousands of nodes. In contrast, the LOUVAIN method exhibited the best memory performance, consuming only 18MB of memory for a network of 100 thousands of nodes. This is due to the fact that the LOUVAIN implementation is based on an internal binary network representation that is much more compact than

the adjacency lists used by the majority of its competing methods. An additional useful observation of how the memory requirements of methods scale with the average degree (i.e. number of edges) in the network can be drawn from Figure 5(h): LOUVAIN and MCL appear relatively insensitive to $\hat{k}$, while the memory requirements for the rest of the methods are heavily dependent on $\hat{k}$.

Our final observations pertain to the precision of the detected community structure, quantified by use of the NMI measure. The number of nodes in the network has only limited effect on the precision of the detected community structure. Only the LOUVAIN and SPIN methods appear to suffer from the increase in the number of nodes; specifically, their NMI performance drops approximately 10% as the number of nodes in the network grows from one to 100 thousands of nodes. Furthermore, the average node degree appears to have a strong impact on the precision of the detected community structure. All algorithms sustain considerable loss in NMI, which is particularly pronounced for MCL (NMI drops below zero after $\hat{k}$ exceeds 50) and LPROP (NMI gets zero once $\hat{k}$ gets 100). The most resilient methods in that respect are WALKTRAP, LOUVAIN and SPIN. Finally, the effect of the mixing parameter on the precision of the detected community structure is in accordance with the observations of Lancichinetti and Fortunato [59], where a sharp drop in NMI is observed as μ increases from 0.5 to 0.8. The most sensitive algorithms to μ are LPROP and CNM, while MCL and WALKTRAP appear to be the most resilient.

4.3 Dynamic community detection

So far, the discussion on community detection has progressed under the silent assumption that the network under consideration is static. In fact, it is only recent works in this area that take the evolving nature of network data into account. Due to the highly dynamic nature of Social Media data, this is a significant aspect of community detection that is worth further attention. In general, time-awareness can be incorporated in community detection approaches in a number of ways: (a) longitudinal application on successive snapshots, (b) vertex-centric time-awareness, (c) incremental application.

Longitudinal application on successive snapshots: This is a simple approach for extending any community detection method for the analysis of dynamic networks. It consists of two basic steps: (a) application of community detection on a sequence of static network snapshots, and (b) identification of correspondence between communities found in successive network snapshots. For instance, this is the approach adopted by Palla et al. [79], where the Clique Percolation Method (Palla et al. [78]) is applied on successive graph snapshots, $G(t)$ and $G(t+1)$, as well as on their union $G(t) \cup G(t+1)$ in order to facilitate the community correspondence identification. An important result of this study pertains to the possible changes that communities undergo throughout their life, e.g. growth, contraction, merging, splitting, birth and death.

Similar community evolution operations are defined in Asur et al. [5], where the MCL graph clustering algorithm (Van Dongen [30]) is used to discover communities on successive graph snapshots. A more recent approach (Kim and Han [54]) relies on the results of SCAN (Xu et al. [110]) on individual graph snapshots and identifies the correspondence between communities in different times by imposing temporal constraints on successive graph snapshots. This is achieved by considering vertex similarity links across graph snapshots at different times and identifying dense t-partite quasi-cliques on the t-partite graphs that are formed by linking similar vertices of successive graph

snapshots. Alternatively, edges between vertices have been used as community features and their correlations across different time snapshots have been used to match and track communities in time (Lin et al. [64]).

Vertex-centric time-awareness: According to this approach, the dynamic character of community structure is attributed to their vertices. For instance, Fenn et al. [33] apply the spin model of Reichardt and Bornholdt [90] on successive graph snapshots to identify communities. Then, for each vertex it quantifies the extent to which the vertex remains in the same community or switches from one community to another. An alternative approach is introduced by Wang et al. [108]: community evolution is described in terms of a small set of *core* vertices. For instance, when two successive communities share a common core vertex, then the second community (from a temporal point of view) is considered as an evolved version of the first. Due to the fact that vertex-centric approaches are also based on successive graph snapshots, they are quite similar to the previous category of dynamic community detection methods.

Incremental application: This is a more sophisticated means of considering the dynamic character of networks. The community detection at time $t + 1$ is initialized with information derived from the community structure discovered at time t. The incremental nature of these methods has two advantages: (a) it is considerably more efficient in terms of computations, (b) it leads to more consistent community detection results, since the clustering at time $t + 1$ is not dramatically different than the one at time t, i.e. a balance is achieved between temporal noise and concept drift (Chakrabarti et al. [19]). Incremental methods can be further divided in iterative approximation schemes (Sun et al. [103], Yang and Liu [111], Lin et al. [65]) and graph modification oriented reclusterings (Falkowski et al. [32], Franke and Geyer-Schulz [39]).

Iterative approximation schemes are inherently incremental since they rely on some iterative scheme that progressively approximates the target community structure. Therefore, in case they are initialized with the community structure found for the previous graph snapshot, they tend to converge to the new community structure in much fewer iterations. For instance, in GraphScope (Sun et al. [103]), community detection is carried out by minimizing the information required to encode a graph and its community structure (Minimum Description Length). Initializing the community structure with some rough estimate (the structure found in the previous timestep) leads to faster minimization of the Minimum Description Length. Similarly, the incremental scheme of Yang and Liu [111] uses an iterative approximation technique for estimating repulsion and attraction forces among vertices that are responsible for separating the network into communities. Finally, FacetNet (Lin et al. [65]) presents another important paradigm of incremental community detection formulated on the basis of *evolutionary clustering* (Chakrabarti et al. [19]). According to it, community detection is formulated as a cost minimization problem that is expressed as the combination of two costs: the *snapshot cost*, which is related to the current network structure, and the *historic cost*, which is derived from the divergence of the current community structure from the previous one. This cost is minimized through an iterative rule update scheme.

The second approach to incremental community detection is based on the idea that the incremental method should only take into account the graph modification operations (e.g. new vertex, new edge, etc.) in order to derive a new community structure, which would be the same or equivalent with the one that would have been derived in case a complete reclustering had taken place. For example, the DenGraph framework (Falkowski et al. [32]), which uses a notion of subgraph density equivalent to the concept of (μ, ϵ)-cores (Xu et al. [110]), translates each modification taking place on the

graph into some update operation (or no operation in some cases) on the graph cores and their associated communities. A more sophisticated scheme is presented by Franke and Geyer-Schulz [39] based on an incremental version of a restricted random walk clustering scheme, which takes into account only the changes that have taken place on the graph since the last clustering.

5 Strategies for scaling community detection on Social Media scale

The remarkable growth of Social Media data has brought attention to the problem of scaling community detection methods to real-world networks that contain many billions of vertices and edges. Here, we present five strategies that can be employed in order to scale community discovery methods to networks of such magnitude.

5.1 Sampling techniques

One possibility for reducing the complexity of community detection methods is to employ a sampling strategy in the computationally expensive part of the method. Previous works (Leskovec and Faloutsos [60], Hübler et al. [47] have demonstrated that it is possible to sample a small subgraph from a large network so that the induced subgraph has the same network properties (e.g. distribution of degree and clustering coefficient) with the original large graph. However, in the context of community detection, vertex (or edge) sampling may have different goals.

For instance, Tyler et al. [106] make use of sampling in the computation of edge betweeness in large graphs, i.e. their sampling targets at approximating the true betweeness of edges. In their betweeness computation, they take into account only the contribution from a limited number of randomly selected vertices, which saves significant computational overhead at a modest statistical error that is further reduced by repeating the sample-based computation several times and aggregating the results. A crude means of approximating community detection is by use of high-degree vertex sampling (Java et al. [51]). Subsequently, the unsampled vertices are assigned to the community to which they have more connections. Following a more refined approach (Maiya and Berger-Wolf [70]), vertices are sampled based on their expansion properties in order to derive a subgraph, of which the community structure is representative and can thus be used to infer the community assignments of unsampled vertices by use of statistical relational learning.

Aside the computational efficiency aspects, sampling network data is important for Social Media data for an additional reason: Any Social Media network under study is essentially the result of a network sampling process that is implemented by some crawling algorithm (Gjoka et al. [44], Ye et al. [113]). For that reason, the properties of the sampled network and the extent to which these are representative of the original network depend on the employed crawling/sampling principle. In the end, the results of community detection will implicitly depend on the adopted network sampling process.

5.2 Local graph processing

Local community detection methods (Flake et al. [34], Clauset [24], Luo et al. [68], Papadopoulos et al. [80], Chen et al. [20]) provide a means to alleviate scalability

challenges by focusing on a portion of the network under study. In practice, such methods start a network exploration process from a seed vertex or vertex set and start attaching adjacent vertices to the community as long as these attachments lead to the increase of some local community quality measure. This principle is illustrated in Figure 6. According to it, the network is divided in four parts: (a) the set B of *border* vertices that are adjacent to at least one vertex not belonging to the community, (b) the set C of *core* community members that have no connection to vertices outside the community, and (c) the set U of vertices that are adjacent to at least one vertex of the community border, (d) the rest of the network. A local method proceeds by examining which vertex (if any) from set U is appropriate for attachment to the community. Once such a vertex is identified and attached to the community, sets B, C, and U are updated and the process continues until it is not possible to identify any other vertex that should be attached to the community.

Since local methods are limited to a portion of the network under study, it is expected that they can circumvent the memory bottleneck faced by global methods. In most local community quality measures, such as local modularity (Clauset [24]), subgraph modularity (Luo et al. [68]) or node outwardness (Bagrow [8]), it is necessary to maintain in memory only the subgraph $G_L = (V_L, E_L) \subseteq G$, where $V_L = C \cup B \cup U$ and $E_L = \{(u, v) \in E | u, v \in V_L\}$. However, in case such a local method leads to the formation of a gigantic community, violating the assumption that $|G_L| \ll |G|$, even local methods are not sufficient to address scalability concerns. Such cases are expected to be rare and dependent on the network topology under study as well as the local community quality measure employed by the method.

Local methods have the added advantage that they are targeted at some specific topic. This is especially important for the Web in general and Social Media applications in particular, since it is not feasible to study the community structure of the whole Web. Usually, a community of interest can be defined by means of some seed items (e.g. web pages, online photos, videos) and then the community containing them can be progressively discovered by use of either their hyperlinks (conventional Web communities (Flake et al. [34], Ino et al. [49]) or by use of the associated metadata (e.g. tags, comments, etc.), which create implicit links between content items.

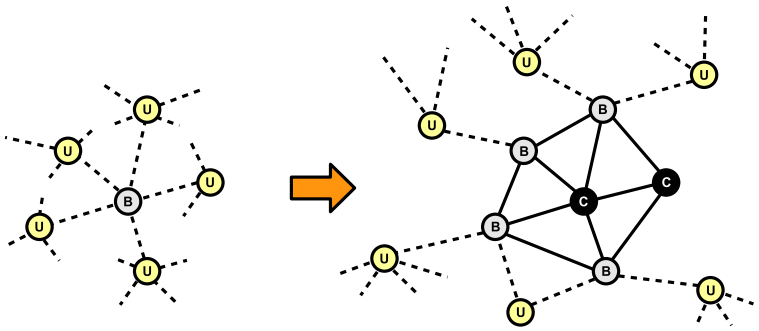


Fig. 6 Basic principle of local community detection: Starting from some seed vertex, one progressively explores the neighborhood around it. Vertices are distinguished between *Core* (C), *Boundary* (B) and *Un-visited* (U), with respect to their relation with the local community.

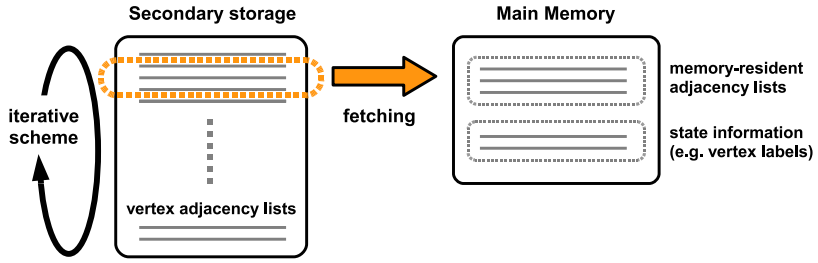


Fig. 7 Basic principle of streaming/iterative schemes: an iterative process visits the vertex adjacency lists and fetches part of them in main-memory, where the community computation is performed by use of the local adjacency information and some additional state variables.

5.3 Streaming/Iterative schemes

Streaming and iterative schemes are a very promising means of scaling community detection. According to them, an iterative process examines each vertex along with its neighbors in a given order and performs a computation, of which the result is associated with the processed vertex. This vertex iteration process is repeated multiple times until convergence. The same principle can be applied by use of edge iteration. In that way, it is possible to perform community detection on networks that do not fit in main memory under the assumption that the relevant parts of the graph (e.g. vertex neighborhoods) are streamed in main memory and state information (e.g. values associated with vertices) resides in main memory. This principle is illustrated in Figure 7.

Label propagation (Raghavan et al. [89]) is a prominent example of the iterative paradigm. According to it, each vertex is visited and assigned a label as a result of majority voting by its neighbors. This vertex iteration is repeated multiple times until convergence.[13] A more complex example of iterative community detection is provided by the (μ, ϵ)-core community detection scheme (Xu et al. [110]) that consists of a single edge and a single vertex iteration: each edge is visited once in order to compute the structural similarity between the vertices it connects. Subsequently, a vertex visiting scheme is carried out that first identifies (μ, ϵ)-cores and then attaches to them those vertices that are structure reachable from them (in total each vertex is visited once).

5.4 Multi-level approaches

Multi-level approaches rely on the following principle: First, they try to find a rough partition of the network into communities by use of a fast process (at the expense of accuracy). Subsequently, the create a meta-network, where vertices represent communities and edges stand for inter-community associations. This meta-network is much smaller in size than the original, therefore more accurate partitioning techniques can be devised. This principle was popularized by the METIS graph partitioning framework (Karypis and Kumar [53]) and applied within a community detection framework by Djidjev [28], where modularity maximization is cast as a minimum weighted cut problem that is solved by recursive bisection and refinement.

[13] Sometimes there may be oscillation phenomena that can be addressed by extensions of the method (Leung et al. [62]).

In addition, a community "folding" or "contraction" process can be integrated in various community detection schemes (Ino et al. [49], Blondel et al. [12]) with the goal of uncovering community organization at multiple resolutions. Ino et al. [49] devise an efficient scheme for enumerating certain types of subgraphs (satisfying the definition of the so-called IKN-community) and apply it recursively at multiple levels in order to uncover hierarchical community structure. Similarly, a heuristic scheme for modularity maximization on large scale networks is presented by Blondel et al. [12]. Once an optimal partition is identified at the lowest resolution, the identified communities are "folded" into vertices and the same scheme is applied at the meta-network comprising communities as vertices. In such methods, the main computational burden falls on the first level of the multi-level scheme. For that reason, heuristics are commonly devised to speed up computations at this level of community detection.

An alternative multi-level scheme was presented by Gibson et al. [42], where large dense bi-cliques of vertices are identified on a directed network. A shingling operation is proposed that maps an input set of size n to a set of much smaller size (a so-called "fingerprint"). This operation is used to create fingerprints of the neighborhoods of vertices (first level shingles) and then re-applied to create second-level shingles. Then, the extraction of dense bi-cliques is translated to a problem of connected component identification on the network of first-level shingles G_S. This problem is much easier than the original due to the fact that it only needs memory access to the vertices of G_S, since its edges are encoded in the second-level shingles. In that way, the authors could analyze the community structure of a network consisting of two billion vertices and 11 billion edges.

5.5 Distribution and parallelization

The constant increase in the size of the networks that need to be analyzed will eventually bring the aforementioned strategies for scaling community detection to their limits assuming they are applied on a single machine. For that reason, an essential strategy for making community detection scalable to peta-scale networks is the distribution of the network into multiple processing nodes and parallel execution of many community detection processes.

Hui et al. [48] present an interesting approach to distributed community detection. Community detection is seen as a local process taking place in each processing node in a network of devices so that each device is "aware" of its community. Although this perspective does not address at all the scalability aspects of the problem, it is interesting for two reasons: (a) it introduces the paradigm of distributed community detection by use of local techniques (subsection 5.2), (b) it demonstrates the practical issues arising when a local community detection method (such as the one based on optimizing the local modularity by Clauset [24]) is applied in an environment where only partial knowledge of the network is possible.

The Bulk Synchronous Model (BSP) is used by Zhang et al. [114] to detect communities in large networks by means of distribution. The processing model is vertex-oriented (similar to the one by Hui et al. [48]) and involves a set of local updates on a local structural property termed "propinquity" and message passing to neighboring vertices. It is demonstrated by authors that this dynamic process leads to the emergence of communities in the form of connected components. The performance of the technique is heavily dependent on the vertex degree distribution, which led the au-

thors to the adoption of some ad hoc degree-based filtering on the network. In terms of parallelism, the authors achieve substantial gains with the addition of new processing nodes up to a meaningful number of nodes given the network size (i.e. it is of no use to partition a relatively small network in many nodes due to the overhead that will be incurred by the excessive message passing).

Alternatively, Yang et al. [112] implement a parallel community detection scheme on top of the *MapReduce* (Dean and Ghemawat [26]) framework. A special mapping and reduction scheme is employed with the goal of distributing the problem of enumerating all maximal cliques in large networks. Subsequently, a clique merging step is performed that results in a clique-based community structure similar to the one discovered by the Clique Percolation Method (Palla et al. [78]), but with no need to specify the k parameter. However, their experiments indicate that after distributing the problem to more than 30 processors, the total processing throughput levels off indicating increased computational overhead and thus calling for improved parallelization mechanisms.

6 Exploitation of community detection results

Community detection has attracted extensive research interest for some time now. Despite the proliferation of methods developed for analyzing different types of communities, there have been relatively less works that actually exploit the results of community detection in some application. Traditionally, social network analysis and biological systems have benefited from the results of community detection (Girvan and Newman [43]). In addition, research on the structure and processes taking place on the Web (Kumar et al. [56], Flake et al. [34]) constitute an additional application domain. It is only recently that some works have started exploiting the results of community detection research on Social Media data. Yet, Social Media present unique opportunities and challenges for community detection research due to the following properties: (a) the unprecedented growth and magnitude of networked data generated by Social Media applications, (b) the high rate of spam and the large variance in the quality of Social Media content and metadata, (c) the highly interactive and dynamic character of Social Media systems.

Here we collect the main applications that community detection has seen to date in the context of Social Media. We differentiate between observational studies that are focused on the structure and dynamics of communities in Social Media networks and application-oriented studies that attempt to integrate community detection results in the context of some information retrieval or web mining task.

6.1 Observational studies

6.1.1 Statistical analysis of Social Media community structure

Several studies focus on the structure of communities in Social Media networks. Most of them consider some specific Social Media application and examine the community structure of the network derived from it. For instance, a study on a collaborative tagging system (delicious) is presented by Cattuto et al. [17], where an online bookmark network is created from the tagging activities of delicious users[14] and a spectral

[14] Two bookmarks are linked by a weighted edge depending on the number of tags they share.

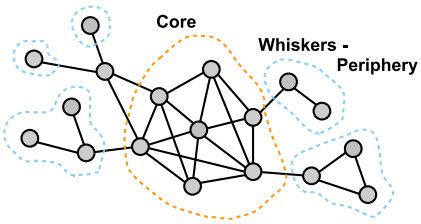


Fig. 8 Core-periphery structure, a characteristic of many Social Media networks [61].

technique is applied for separating the resources into communities. Another study (Zakharov [115]) focuses on the community structure of the users of the LiveJournal blogging platform. Through a thermodynamic diffusion process on the network, the authors could identify several prominent communities on the network (e.g. a russian-speaking blogging community and two communities focused on two RPGs). Another study on blog communities is presented by Java et al. [50]; in that, a co-clustering framework is employed in order to exploit both the hyperlink structure of blog articles as well as the tag information of articles.

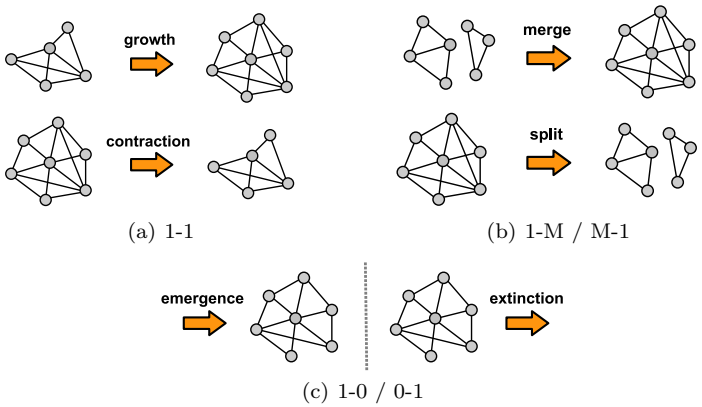
One of the most extensive and systematic studies on the community structure of networks, including several Social Media ones, is presented by Leskovec et al. [61]. The examined Social Media networks are mostly user affiliation networks from different Social Media applications (Delicious, Epinions, Flickr, LinkedIn, LiveJournal, Yahoo! Answers), but there is also an interesting network comprising Netflix users and movies as vertices and ratings as edges (bipartite). The authors employ both a hybrid graph partitioning technique based on METIS (Karypis and Kumar [53]) and MQI (Gallo et al. [40]), and a local spectral partitioning technique (Andersen et al. [2]), both of which identify partitions with minimum conductance (Kannan et al. [52])[15]. In order to express the community structure quality in different scales, the authors employ the Network Community Plot, i.e. a plot of the minimum conductance value in relation to the community of size.

From the analysis across a wide range of networks and by use of different graph partition algorithms, it appears that the communities of the networks of interest are integrated in a core-periphery structure, where the periphery consists of the so-called "whiskers", densely connected subgraphs (of size in the order of 100 vertices) that are connected to the rest of the network by only few (typically one or two) edges. A pictorial representation of this structural pattern is depicted in Figure 8. The study revealed that such peripheral communities can collectively constitute even more than half the size of the network and are responsible for the "best" conductance values achieved by graph partitioning schemes. As communities get larger, they tend to blend in with the core of the network and it is harder to separate them by means of graph partitioning.

6.1.2 Temporal analysis of Social Media communities

Thanks to the recent development of numerous methods for discovering and dynamically tracking the community structure of networks (subsection 4.3), the focus of community detection studies on Social Media has begun to shift from static community

[15] The reason for using two methods is that the global one focuses on minimizing the conductance over the whole graph, while the second also makes sure that local community properties, such as compactness, are preserved even at the expense of conductance minimization.



(a) 1-1 (b) 1-M / M-1

(c) 1-0 / 0-1

Fig. 9 Basic primitives of community structure evolution: (a) growth and contraction, (b) merge and split, (c) emergence and extinction.

structure analysis to their temporal evolution and dynamics. Consequently, there have been several efforts to understand the evolving phenomena that characterize Social Media communities. For instance, Lin et al. study the evolution of blog communities forming around real-world events by analyzing the temporal correlations of interactions between bloggers of the same community through observation of their mutual links [64]. Schlitter and Falkowski study the community dynamics of an online music listening application (last.fm) by use of their dynamic community detection framework [95].

The aforementioned works observed the formation of new communities, the evolution (growth or contraction) of persistent ones, as well as more complicated community dynamics (merging, splitting). Comparing these studies with the seminal work by Palla et al. [79], it appears that some consensus is formed regarding the possible transformations that communities may undergo. Figure 9 illustrates six basic transformations that have been identified in a number of studies (Lin et al. [64], Palla et al. [79], Schlitter and Falkowski [95]). Basically, there are three types of transformations: (a) one-to-one (subfigure 9(a)), which involves community growth or contraction, (b) one-to-many (subfigure 9(b)), which involves one community splitting to many or many communities merging to one, and (c) one-to-zero (subfigure 9(c)), which involves the emergence or the extinction of a community.

6.2 Applications

6.2.1 Topic detection in collaborative tagging systems

The huge amount of tags attached to online content by users of Social Media applications creates the need for imposing organization on the flat tag spaces of collaborative tagging applications. This can be directly achieved by grouping tags based on the topic they are associated with. There have been several recent works that attempt to derive meaningful clusterings of tags that correspond to topics of social interest (Begelman et al. [11], Papadopoulos et al. [80–83], Simpson [101]). For instance, Begelman et al. [11] were among the first to apply community detection methods, namely spectral

modularity maximization, to identify interesting tag clusters. Similarly, tag clustering is pursued by means of a variant of the modularity maximization method of Newman [74] on enterprise folksonomies (Simpson [101]). A problem that the latter study pinpointed is that the employed modularity maximization method led to the discovery of disproportionately large communities when the underlying tag co-occurrence network was dense (e.g. the tag network formed from delicious data).

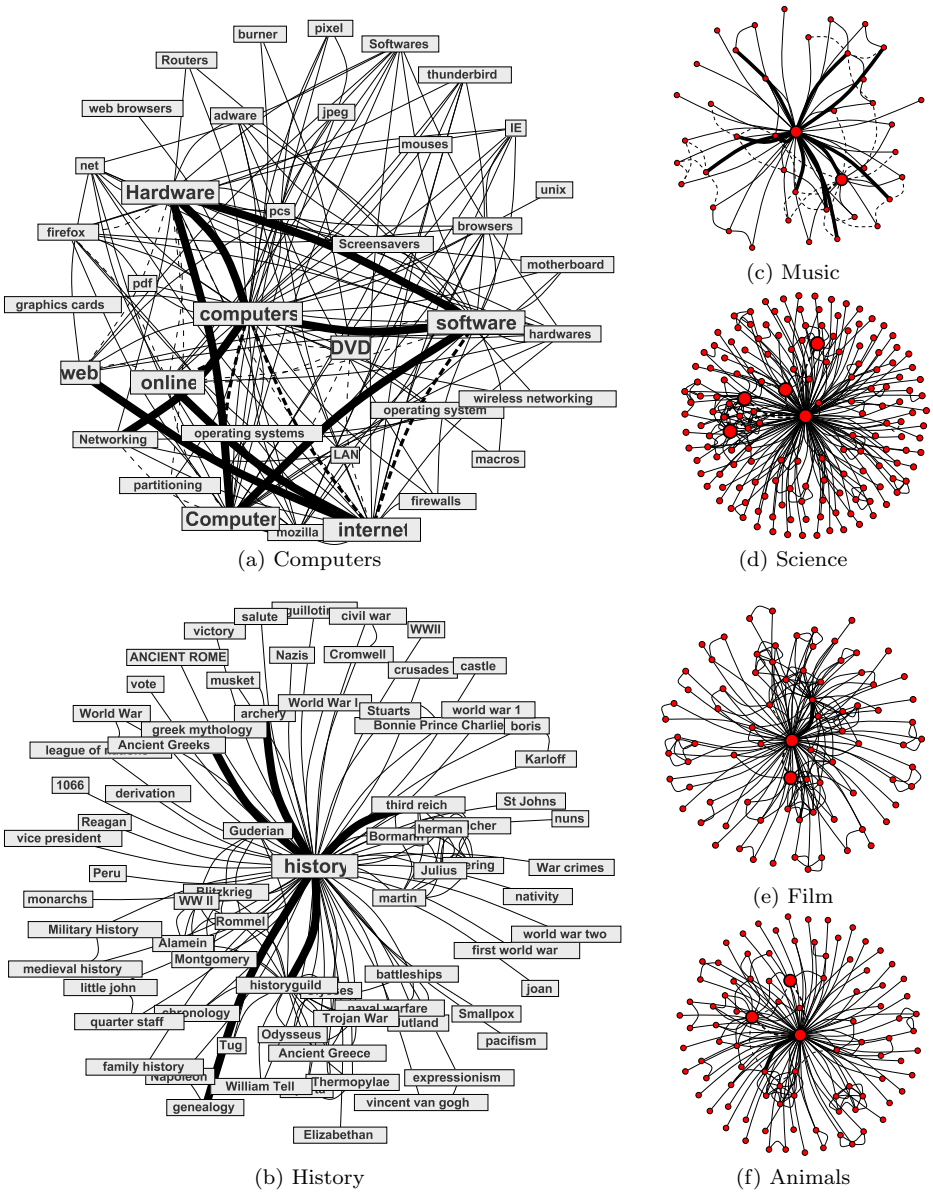
Alternative tag community detection schemes were employed by Papadopoulos et al. [80–82]. In the first work [80], an efficient local community detection scheme was devised that could discover the community around a seed tag. Figure 10 presents several examples of tag communities discovered on a tag network created from the LYCOS iQ community question answering application (similar to Yahoo! Answers). Such communities can contribute to understanding the interests of the users of Social Media applications. For instance, by observing the community structure for the topics Music (subfigure 9(c)) and Science (subfigure 9(d)), it becomes obvious that in LYCOS iQ there is more interest and a more sophisticated topic structure with respect to the topic of Science than to Music.

An alternative tag community detection scheme was presented in in [81]. The proposed scheme relied on a two-step process that involved the selection of densely connected groups of tags by use of SCAN (Xu et al. [110]) and the expansion of these groups with the goal of maximizing the subgraph modularity of Luo et al. [68], which could be efficiently computed in an incremental fashion. A refinement of this method appeared in [82] solving the problem of parameter selection introduced by the first step of the algorithm. A limitation of this approach was the *topic blending* problem, i.e. the situation in which some tag communities contained tags related to different topics due to some polysemous tag that acted as bridge between them. This has been addressed by tag disambiguation methods that are discussed below.

6.2.2 Tag disambiguation

Due to the unrestricted and informal nature of tagging, there are numerous cases where the use of a single tag in isolation is not sufficient to convey the intended semantics. For that reason, tags need to be considered in context in order to disambiguate their meaning. Recently, several research efforts (Au Yeung et al. [6], Specia and Motta [102]) attempted to address the problem of tag disambiguation by use of community detection. Starting from a particular tag, Au Yeung et al. [6] derive several Social Media networks, e.g. a network of documents that have been tagged with the particular tag by the same user, and the community detection method of Newman [76] is applied to extract communities of tags or documents (that eventually lead to tags) that correspond to the different senses of a tag. This approach was demonstrated to yield superior performance compared to consulting some static external source of information such as WordNet.

Specia and Motta [102] employ tag clustering within a tag-to-ontology mapping framework. They achieve this by first clustering tags through a heuristic scheme similar to seed-based local community expansion (as described in subsection 5.2) in order to identify tag communities around pairs of tags that co-occur frequently with each other. This tag clustering scheme is extended by a cluster post-processing step to remove redundant clusters and the resulting clusters are used for mapping tags to ontology concepts and relations, as well as to Wikipedia entries. Experimental results on Flickr and delicious demonstrated the feasibility of this approach to semantify user-contributed content.



(a) Computers

(b) History

(c) Music

(d) Science

(e) Film

(f) Animals

Fig. 10 Examples of tag communities discovered by the local method of [80]. The presented communities were created using "computers", "history", "music", "science", "film" and "animals" as seed nodes. For subfigures 9(a) and 9(b), the tags are depicted. For the sake of brevity, only the network structure is illustrated for the rest of the cases. Edges are weighted by cooccurrence frequency and only those exceeding a certain threshold are drawn for clarity.

6.2.3 User profiling

Personalized search and recommendation constitute an additional information retrieval problem that can benefit from the use of community detection. More specifically, clusters of tags have been demonstrated to act as effective proxies of users' interests. Gemell et al. [41] base the ranking performance of a personalized search mechanism on tag clusters outperforming conventional ranking schemes. The tag clusters were extracted by use of a variant of hierarchical agglomerative clustering. However, since this scheme requires manual parameter tuning that may have significant impact on performance (as remarked by the authors), a viable alternative would be to use some community detection to identify tag clusters.

Tsatsou et al. [105] integrate the results of tag community detection in a personalized ad recommendation system and compared against conventional nearest-neighbor tag expansion schemes. More specifically, tags belonging to the same community are used as a terminological description of semantic concepts within a domain ontology. It was found that the use of tag communities discovered by a local community detection algorithm (Papadopoulos et al. [81]) led to increased recall performance and faster convergence of the personalized profiles. This is due to the fact that the rich associations among tags that emerge through the identified community structure may partly alleviate the *vocabulary impedance* problem that is known to trouble online advertising (Ribeiro-Neto et al. [91]).

6.2.4 Photo clustering

The huge increase in the amounts of user contributed content in Social Media sharing applications such as Flickr creates the need for automatically organizing content. For instance, clustering the photos in such an application may help users to navigate larger parts of the photo collection more efficiently (i.e. by looking at one representative photo per cluster instead of all cluster photos). Moëllic et al. [73] pursue photo clustering by use of a shared nearest neighbors approach on two graphs of photos where edges between photos are considered either by use of shared tags (tag-based graph) or due to visual similarity (visual graph). The employed clustering technique is shown to achieve improved clustering performance compared with conventional clustering algorithms (k-means and one of its speeded-up variants). Also, comparing the results of their methods with the clusters available from Flickr, the authors noted similar clustering quality.

A more sophisticated application of graph clustering is presented by Li et al. [63]. Their goal is to collect different representative (*iconic*) photos for popular landmarks and use the massive visual content that is associated with them in order to create 3D landmark models. They devise a multi-stage photo processing framework, in which an important task is to group iconic photos together in order to reduce the amount of photos that are processed by the computationally intensive 3D reconstruction step. They achieve photo grouping by creating an iconic photo graph where photos are connected by edges when they are visually similar (by use of SIFT descriptors) and by applying the N-cut graph clustering algorithm by Shi and Malik [99] on this graph. In that way, they managed to reconstruct the major views of three famous landmarks (Statue of Liberty, Notre Dame and San Marco).

Papadopoulos et al. [84] identify real-world landmarks and events in large tagged photo collections by use of photo cluster classification. They apply the SCAN algorithm (Xu et al. [110]) on a hybrid photo similarity graph that encodes both visual and tag

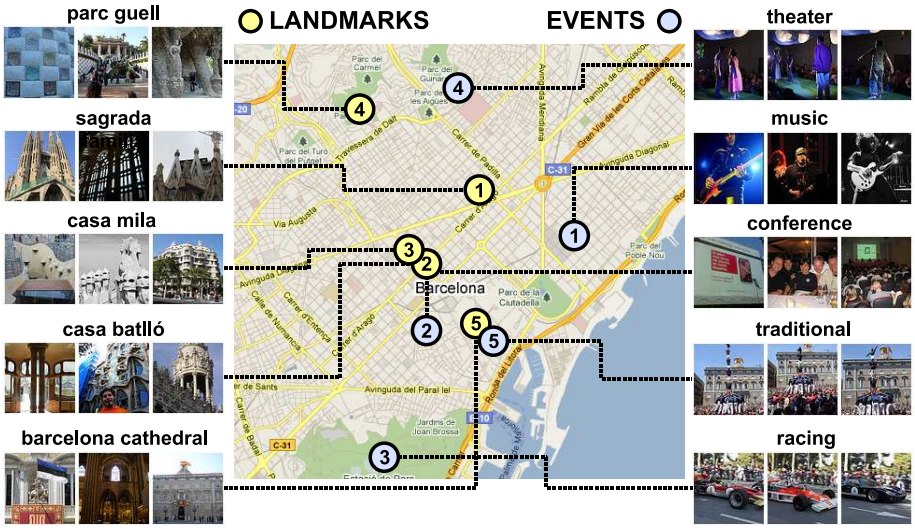


Fig. 11 Five sample landmarks and events in Barcelona as identified by the cluster-based landmark and event detection framework of [84]. Thanks to the geotagging information of the photos, the landmarks and events can be localized with high precision.

similarity between photos. Subsequently, the derived photo and tag communities are classified as landmarks or events based on cluster features similar to the ones used by Quack et al. [87]. Figure 11 illustrates five sample landmarks and event found for the city of Barcelona by use of this method. Manual inspection of the results reveals that most of the clusters correspond either to famous landmarks of the city or to real events (e.g. music concerts). Furthermore, the automatically selected cluster tags provide meaningful descriptions for them.

6.2.5 Event detection

Events constitute an important unit of organization for Social Media content, since a large part of user contributed content revolves around real-world events. Community detection has found applications in the detection and tracking of events from social text streams. For instance, the framework presented by Zhao et al. [116] incorporates textual, social and temporal aspects of blog feeds with the goal of tracking events. The N-cut graph partitioning method of Shi and Malik [99] is used twice in this framework: once to cluster a graph of blog posts connected by their textual similarity into topics, and at a second level, to cluster a graph of temporal activity profiles among users (created by their comments) into communities that correspond to real-world events. They tested their approach on a sample of 250 thousand posts from the Dailykos political blog, from which they could extract well-recognizable real-world events.

An even simpler approach for the extraction of events from blog streams is described by Sayyadi et al. [93]. A keyword graph is built by extracting important keywords (named entities, noun phrases) from text documents and associating them through their co-occurrence. Then, the authors employ the divisive community detection approach of Girvan and Newman [43] with a slight variation that enables overlap between communities: keywords that are estimated to belong to many communities are

split into multiple vertices. They also use the efficient sample-based version of the algorithm (Tyler et al. [106]) and observe that the derived communities are the same. Subsequently, they use the extracted keyword communities as prototype vectors for clustering the news articles. With the help of some additional cluster filtering and merging, they end up with document clusters corresponding to real-world events.

7 Conclusions

7.1 Summary of main findings

The paper provided an overview of the emerging topic of community detection on Social Media with a focus on scalability aspects and applications. Due to the huge number of related works on the topic, the discussion focused on the most established notions of community on networks and widely used methodological paradigms of community detection, including their performance characteristics in terms of complexity and memory usage. Furthermore, several strategies were presented that can be used by community detection frameworks for scaling their applicability to real-world datasets. Finally, the paper presented a series of observational studies and applications, where the results of community detection have been exploited.

A first conclusion pertains to the very concept of community. There are numerous network-based definitions and perceptions of community. In the context of Social Media, communities comprise entities, such as online users, content items and metadata associated with content, that revolve around particular topics or events of social interest. The association of such entities through their contextual co-occurrence leads to the formation of Social Media networks. Analyzing the structure of such networks leads to the discovery of Social Media communities.

Due to the graph partitioning heritage of community detection, many of the available definitions are based on a global network perspective, i.e. they consider community structure as a property of the whole network. However, a local perspective on the problem may be more meaningful in the context of Social Media, since one can only have partial knowledge of the network under study. In addition, the possibility of overlap among Social Media communities and weighted membership to them are especially important attributes. Finally, the support for leaving noisy data out of the discovered community structure and considering multiple levels of community organization constitute significant aspects of the problem in this domain.

In addition, this survey acknowledged the remarkable proliferation of community detection methods developed in the last years. Due to the diversity of research disciplines spawning the development of these methods, namely social network analysis, computer science and statistical physics, there is no widely accepted terminology and methodological classification of existing methods. For that reason, we attempted to provide such a classification in Section 4.1. Despite the large number of community detection methods abounding in literature, in practice most applications are limited to a few seminal ones (or variants of them): the divisive scheme by Girvan and Newman [43], the modularity maximization method of Newman [74] and its computationally efficient implementation (Clauset et al. [23]), the N-cut graph partitioning scheme by Shi and Malik [99], and spectral partitioning by use of the graph Laplacian (Von Luxburg [69]). However, as it became clear from the performance analysis of subsec-

tion 4.2, these methods are not the most efficient, neither in terms of complexity nor in terms of memory usage.

We believe that there are some under-exploited categories of community detection methods. For instance, local density-based schemes, such as the SCAN algorithm (Xu et al. [110]), are particularly important in the context of Social Media due to their support for leaving spuriously connected vertices (i.e. noise) out of the detected community structure. In addition, iterative dynamic processes, such as the label propagation algorithm by Raghavan et al. [89] are also promising due to their computational efficiency and their conceptual simplicity, which facilitates the development of method extensions or adaptations that are catered for particular problems.

A final note concerns the relation of community detection to the emergence of core-periphery structure in Social Media networks. While acknowledging the existence of such structure from a macroscopic perspective, one should understand its implications on the exploitation of community detection in Social Media mining. Established community detection methods that rely on the optimization of some quality measure (modularity maximization, conductance/N-cut minimization) result in the extraction of disproportionately large communities from the core of Social Media networks. Such gigantic communities are of limited use within information retrieval tasks, such as online content recommendation and retrieval. Therefore, we deem that appropriate methods capable of extracting fine-grained community structure within such gigantic components are the most relevant in the context of Social Media mining applications.

7.2 Outlook

Given the growing interest in the problem of community detection and the paramount importance of Social Media for business and society in whole, it is only natural to expect that exciting developments will take place on this field in the years to come. We foresee significant progress on several methodological issues faced by existing methods, but most importantly we expect that community detection results will be further exploited in several online scenarios and will attain a sufficiently mature state in order to be deployed in real-world Social Media applications.

On a methodological level, one should expect further refinement of methods to cope with the increasing scale of Social Media data. Method parallelization and fingerprinting or summarization techniques are deemed as promising strategies to achieve scalability. Moreover, the development of new methods that will enable the study of k-partite, multi-relational and hyper networks is likely to attract considerable interest. Currently, there is a clear preference towards applying community detection on simple one-mode networks (even when it is necessary to employ some lossy transformation on the original data) due to the conceptual and computational simplicity of these methods. It remains to be seen whether such methods will be replaced by more sophisticated that are applicable directly to the "raw" (implying k-partite, multi-relational) form of Social Media networks.

In terms of application, one should expect that recent community detection methods will displace conventional clustering and partitioning schemes, such as k-means and hierarchical agglomerative clustering, which are still very popular in a series of domains, e.g text and photo clustering. In addition, as noted above, even within the space of community detection methods, one should expect a shift in their adoption: the currently established global optimization algorithms will give their place to local

density-based methods and iterative approximation schemes. Finally, one may look forward to the wide adoption of dynamic community detection and tracking methods in real-world contexts and applications. In conclusion, the forthcoming developments in the field of community detection are expected to have a transformational impact on mining and understanding the realm of Social Media.

Acknowledgements This work was supported by the WeKnowIt and GLOCAL projects, partially funded by the European Commission, under contract numbers FP7-215453 and FP7-248984 respectively.

References

1. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G. (2008) Finding high-quality content in social media. Proceedings of WSDM '08: the international Conference on Web Search and Web Data Mining (Palo Alto, California, USA, February 11 - 12, 2008), ACM, New York, NY, 183-194
2. Andersen, R., Chung, F.R.K., Lang, K. (2006) Local graph partitioning using PageRank vectors. In FOCS 06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, pages 475486
3. Arenas, A., Díaz-Guilera, A., Pérez-Vicente, C.J. (2006) Synchronization reveals topological scales in complex networks, Physical Review Letters 96 (11), 114102
4. Arenas, A., Duch, J., Fernaández, A., Gómez, S. (2007) Size reduction of complex networks preserving modularity. New Journal of Physics 9, 176
5. Asur, S., Parthasarathy, S., Ucar, D. (2007) An event-based framework for characterizing the evolutionary behavior of interaction graphs. In Proceedings of the 13th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (San Jose, California, USA, August 12 - 15, 2007). KDD '07. ACM, New York, NY, 913-921
6. Au Yeung, C.M., Gibbins, N., Shadbolt., N. (2009) Contextualising Tags in Collaborative Tagging Systems. Proceedings of ACM Conference on Hypertext and Hypermedia, 251-260
7. Baeza-Yates, R. (2007) Graphs from search engine queries. Theory and Practice of Computer Science (SOFSEM), LNCS 4362: 18, Harrachov, Czech Republic, Springer
8. Bagrow J.P. (2008) Evaluating local community methods in networks. Journal of Statistical Mechanics 5, P05001
9. Barber, M.J. (2007) Modularity and community detection in bipartite networks. Physical Review E 76, 066102
10. Batagelj, V., Zaversnik, M. (2003) An O(m) Algorithm for Cores Decomposition of Networks. Eprint arXiv:cs/0310049
11. Begelman, G., Keller, P., Smadja, F. (2006) Automated Tag Clustering: Improving search and exploration in the tag space. Online article: `http://www.pui.ch/phred/automated_tag_clustering`
12. Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E. (2008) Fast unfolding of communities in large networks. Eprint arXiv:0803.0476
13. Borgatti, S., Everett, M., Shirey, P. (1990) LS sets, lambda sets, and other cohesive subsets. Social Networks 12, 337-358
14. Breiger, R., Boorman, S., Arabie, P. (1975) An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. Journal of Mathematical Psychology 12: 328-383
15. Bron, C., Kerbosch, J. (1973) Algorithm 457: finding all cliques of an undirected graph. Communications of the ACM 16 (9): 575577
16. Cattuto, C., Benz, D., Hotho, A., Stumme, G. (2008) Semantic Grounding of Tag Relatedness in Social Bookmarking Systems. Proccedings of ISWC 2008, Karlsruhe, Germany
17. Cattuto, C., Baldassarri, A., Servedio, V.D.P., Loreto, V. (2008) Emergent Community Structure in Social Tagging Systems. Advances in Complex Systems (ACS), 11(4), p. 597-608
18. Chakrabarti, D. (2004) Autopart: Parameter-free graph partitioning and outlier detection. In Lecture Notes in Computer Science 3202, Springer, 112-124
19. Chakrabarti, D., Kumar, R., Tomkins, A. (2006) Evolutionary clustering. In Proceedings of the 12th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (Philadelphia, PA, USA, August 20 - 23, 2006). KDD '06. ACM, New York, NY, 554-560

20. Chen, J., Zaiane, O. R., Goebel, R. (2009) Local Community Identification in Social Networks. International Conference on Advances oin Social Networks Analysis and Mining (ASONAM), Athens, Greece
21. Chen, J., Zaiane, O. R., Goebel, R. (2009) A Visual Data Mining Approach to Find Overlapping Communities in Networks. International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Athens, Greece
22. Chi, Y., Zhu, S., Hino, K., Gong, Y., Zhang, Y. (2009) iOLAP: A framework for analyzing the internet, social networks, and other networked data. Transactions on Multimedia 11(3), 372-382
23. Clauset, A., Newman, M.E.J., Moore, C. (2004) Finding community structure in very large networks. Physical Review E, 70
24. Clauset, A. (2005) Finding local community structure in networks. Physical Review E 72, 026132
25. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A. (2005) Comparing community structure identification. Journal of Statistical Mechanics P09008
26. Dean, J., Ghemawat, S. (2004) Mapreduce: Simplified data processing on large clusters. Proceedings of OSDI 04, pages 137150
27. Dhillon, I. S., Guan, Y., Kulis, B. (2007) Weighted Graph Cuts without Eigenvectors: A Multilevel Approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(11), 1944-1957.
28. Djidjev, H. N. (2008) A Scalable Multilevel Algorithm for Graph Clustering and Community Structure Detection. Lecture Notes In Computer Science, vol. 4936. Springer-Verlag, Berlin, Heidelberg, 117-128
29. Donetti, L, Munoz, M.A. (2004) Detecting network communities: A new systematic and efficient algorithm. Journal of Statistical Mechanics P10012
30. Van Dongen, S. (2000) Graph clustering by flow simulation. Ph.D. Thesis, Dutch National Research Institute for Mathematics and Computer Science, Utrecht, Netherlands
31. Duch, J., Arenas, A. (2005) Community detection in complex networks using extremal optimization. Physical Review E, 72
32. Falkowski, T., Barth, A., Spiliopoulou, M. (2007) DENGRAPH: A Density-based Community Detection Algorithm. Proceedings of Web Intelligence 2007, pp. 112-115
33. Fenn, D., Porter, M., McDonald, M., Williams, S., Johnson, N., Jones, N. (2009) Dynamic communities in multichannel data: An application to the foreign exchange market during the 2007–2008 credit crisis. Eprint arXiv: 0811.3988
34. Flake, G. W., Lawrence, S., Giles, C. L. (2000) Efficient identification of Web communities. Proceedings of KDD '00, ACM, pp. 150-160
35. Fortunato, S., Latora, V., Marchiori, M. (2004) Method to find community structures based on information centrality. Physical Review E 70, 056104
36. Fortunato, S., Castellano, C. (2007) Community Structure in Graphs. Eprint arXiv:0712.2716
37. Fortunato, S. (2009) Community detection in graphs. Eprint arXiv:0906.0612
38. Fortunato S. (2010) Community detection in graphs. Physics Reports 486: 75-174
39. Franke, M., Geyer-Schulz, A. (2009) An update algorithm for restricted random walk clustering for dynamic data sets. Advances in Data Analysis and Classification 3(1): 63-92
40. Gallo, G., Grigoriadis, M.D., Tarjan, R.E. (1989) A fast parametric maximum flow algorithm and applications. SIAM Journal on Computing, 18(1):3055
41. Gemmell, J., Shepitsen A., Mobasher B., Burke, R. (2008) Personalizing Navigation in Folksonomies Using Hierarchical Tag Clustering. Proceedings of DaWaK 2008, LNCS 5182: 196-205
42. Gibson, D., Kumar, R., Tomkins, A. (2005) Discovering large dense subgraphs in massive graphs. In Proceedings of the 31st international Conference on Very Large Data Bases (Trondheim, Norway, August 30 - September 02, 2005). Very Large Data Bases. VLDB Endowment, 721-732
43. Girvan, M., Newman, M.E.J. (2002) Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99 (12): 7821-7826
44. Gjoka, M., Kurant, M., Butts, C. T., Markopoulou, A. (2009) A Walk in Facebook: Uniform Sampling of Users in Online Social Networks. Eprint arXiv: 0906.0060
45. Gregory, S. (2009) Finding overlapping communities in networks by label propagation. Eprint arXiv: 0910.5516
46. Hastings, M.B. (2006) Community detection as an inference problem. Physical Review E 74, 035102

47. Hübler, C., Kriegel, H., Borgwardt, K., Ghahramani, Z. (2008) Metropolis Algorithms for Representative Subgraph Sampling. In Proceedings of the 2008 Eighth IEEE international Conference on Data Mining (December 15 - 19, 2008). ICDM. IEEE Computer Society, Washington, DC, 283-292
48. Hui, P., Yoneki, E., Chan, S. Y., and 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. ACM, pp. 1-8
49. Ino, H., Kudo, M., and Nakamura, A. (2005) Partitioning of Web graphs by community topology. Proceedings of the 14th international Conference on World Wide Web (Chiba, Japan, May 10 - 14, 2005). WWW '05. ACM, New York, NY, 661-669
50. Java, A., Joshi, A., Finin, T. (2008) Detecting Commmunities via Simultaneous Clustering of Graphs and Folksonomies. In Proceedings of WebKDD 2008, KDD Workshop on Web Mining and Web Usage Analysis, Las Vegas, NV
51. Java, A., Joshi, A., Finin, T. (2008) Approximating the Community Structure of the Long Tail. Proceedings of the International Conference on Weblogs and Social Media
52. Kannan, R., Vempala, S., Vetta, A. (2004) On clusterings: Good, bad and spectral. Journal of the ACM 51, 3: 497515
53. Karypis, G. Kumar, V. (1998) A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. SIAM J. Sci. Comput. 20, 1 (Dec. 1998), 359-392
54. Kim, M., Han, J. (2009) A particle-and-density based evolutionary clustering method for dynamic networks. Proceedings of the VLDB Endowment 2(1), 622-633
55. Kovács, I.A., Palotai, R., Szalay, M.S., Csermely, P. (2010) Community Landscapes: An Integrative Approach to Determine Overlapping Network Module Hierarchy, Identify Key Nodes and Predict Network Dynamics. PLoS ONE 5(9): e12528
56. Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. (1999) Trawling the Web for emerging cyber-communities. Computer Networks 31(11-16): 1481-1493
57. Kumar, S.R., Raghavan, P., Rajagopalan, S., Sivakumar, D., Tomkins, A., Upfal, E. (2000) The Web as a Graph. ACM Symposium on Principles of Database Systems (Dallas, Texas)
58. Lancichinetti, A., Fortunato, S., Radicchi, F. (2008) Benchmark graphs for testing community detection algorithms. Physical Review E 78, 046110
59. Lancichinetti, A., Fortunato, S. (2009) Community detection algorithms: A comparative analysis. Physical Review E 80, 056117
60. Leskovec, J., Faloutsos, C. (2006) Sampling from large graphs. In Proceedings of the 12th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (Philadelphia, PA, USA, August 20 - 23, 2006). KDD '06. ACM, New York, NY, 631-636
61. Leskovec, J., Lang, K., Dasgupta, A., Mahoney, M. (2008) Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. Eprint arXiv: 0810.1355
62. Leung, I. X. Y., Hui, P., Lio, P., Crowcroft, J. (2009) Towards real-time community detection in large networks. Physical Review E 79, 066107
63. Li, X., Wu, C., Zach, C., Lazebnik, S., Frahm, J. (2008) Modeling and Recognition of Landmark Image Collections Using Iconic Scene Graphs. Lecture Notes In Computer Science, vol. 5302. Springer-Verlag, Berlin, Heidelberg, 427-440
64. Lin, Y., Sundaram, H., Chi, Y., Tatemura, J., Tseng, B. L. (2007) Blog Community Discovery and Evolution Based on Mutual Awareness Expansion. In Proceedings of the IEEE/WIC/ACM international Conference on Web intelligence. IEEE Computer Society, Washington, DC, 48-56
65. Lin, Y., Chi, Y., Zhu, S., Sundaram, H., Tseng, B. L. (2008) Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In Proceeding of the 17th international Conference on World Wide Web (Beijing, China, April 21 - 25, 2008). WWW '08. ACM, New York, NY, 685-694
66. Lin, Y., Sun, J., Castro, P., Konuru, R., Sundaram, H., and Kelliher, A. (2009) MetaFac: community discovery via relational hypergraph factorization. Proceedings of KDD '09, ACM, pp. 527-536
67. Lorrain, F., White, H. (1971) Structural equivalence of individuals in social networks. Journal of Mathematical Sociology 1, 49-80
68. Luo, F., Wang, J. Z., Promislow, E. (2006) Exploring Local Community Structures in Large Networks. Proceedings of Web Intelligence 2006. IEEE Computer Society, pp. 233-239
69. Von Luxburg, U. (2006) A tutorial on spectral clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, August 2006

70. Maiya, A. S., Berger-Wolf, T. Y. (2010) Sampling community structure. In Proceedings of the 19th international Conference on World Wide Web (Raleigh, North Carolina, USA, April 26 - 30, 2010). WWW '10. ACM, New York, NY, 701-710
71. Massen, C.P., Doye, J. P. K. (2005) Identifying "communities" within energy landscapes. Physical Review E, 71
72. Mika, P. (2005) Ontologies Are Us: A Unified Model of Social Networks and Semantics. Proceedings of ISWC 2005, Springer Berlin/Heidelberg, 522-536
73. Moëllic, P., Haugeard, J., Pitel, G. (2008) Image clustering based on a shared nearest neighbors approach for tagged collections. In Proceedings of CIVR '08 (Niagara Falls, Canada, July 7 - 9) ACM, New York, NY, 269-278
74. Newman, M.E.J., Girvan, M. (2004) Finding and evaluating community structure in networks. Physical Review E 69, 026113
75. Newman, M.E.J. (2004) Fast algorithm for detecting community structure in networks. Physical Review E 69, 066133
76. Newman, M.E.J. (2004) Analysis of weighted networks. Physical Review E 70, 056131
77. Newman, M.E.J. (2006) Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74
78. Palla, G., Derenyi, I., Farkas, I., Vicsek, T. (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature, 435(7043): 814-818
79. Palla, G., Barabasi, A.-L., Vicsek, T. (2007) Quantifying social group evolution. Nature 446: 664-667
80. Papadopoulos, S., Skusa, A., Vakali, A., Kompatsiaris, Y., Wagner, N. (2009) Bridge Bounding: A Local Approach for Efficient Community Discovery in Complex Networks. Eprint arXiv: 0902.0871
81. Papadopoulos, S., Kompatsiaris, Y., Vakali, A. (2009) Leveraging Collective Intelligence through Community Detection in Tag Networks. In Proceedings of CKCaR'09 Workshop on Collective Knowledge Capturing and Representation, Redondo Beach, California, USA
82. Papadopoulos, S., Kompatsiaris, Y., Vakali, A. (2010) A Graph-based Clustering Scheme for Identifying Related Tags in Folksonomies. In Proceedings of DaWaK'10 (Bilbao, Spain), Springer-Verlag, 65-76
83. S. Papadopoulos, A. Vakali, Y. Kompatsiaris (2010) Community Detection in Collaborative Tagging Systems. In Book Community-built Database: Research and Development, Springer
84. Papadopoulos, S., Zigkolis, C., Kompatsiaris, Y., Vakali, A. (2010) Cluster-based Landmark and Event Detection on Tagged Photo Collections. IEEE Multimedia Magazine
85. Pons, P., Latapy, M. (2005) Computing Communities in Large Networks Using Random Walks. Computer and Information Sciences - ISCIS 2005
86. Porter, M.A., Onnela, J.P., Mucha, P.J. (2009) Communities in networks. Notices of the American Mathematical Society, 56(9):1082-1097
87. Quack, T., Leibe, B., Van Gool, L. (2008) World-scale mining of objects and events from community photo collections. In Proceedings of the 2008 international Conference on Content-Based Image and Video Retrieval (Niagara Falls, Canada, July 07 - 09, 2008). CIVR '08. ACM, New York, NY, 47-56
88. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D. (2004) Defining and identifying communities in networks. Proceedings of the National Academy of Sciences 101, 2658-2663
89. Raghavan U. N., Albert, R., Kumara, S. (2007) Near linear time algorithm to detect community structures in large-scale networks. Physical Review E 76, 036106 (2007)
90. Reichardt, J., Bornholdt, S. (2006) Statistical mechanics of community detection. Physical Review E 74, 016110
91. Ribeiro-Neto, B., Cristo, M., Golgher, P. B., Silva de Moura, E. (2005) Impedance coupling in content-targeted advertising. In Proceedings of the 28th Annual international ACM SIGIR Conference (Salvador, Brazil, August 15 - 19). SIGIR '05. ACM, New York, NY, 496-503
92. Rosvall, M., Bergstrom, C.T. (2008) Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences 105, 1118-1123
93. Sayyadi, H., Hurst, M., Maykov, A. (2009) Event Detection and Tracking in Social Streams. Proceedings of International AAAI Conference on Weblogs and Social Media, AAAI Press
94. Schaeffer, S.E. (2007) Graph clustering. Computer Science Review, 1(1): 27-64
95. Schlitter, N., Falkowski, T. (2009) Mining the Dynamics of Music Preferences from a Social Networking Site. Proceedings of the International Conference on Advances in Social Network Analysis and Mining (Athens, Greece)

96. Schmitz, C., Hotho, A., Jäschke, R., Stumme, G. (2006) Mining Association Rules in Folksonomies. Data Science and Classification: 261-270
97. Scott, J. (2000) Social Network Analysis: A Handbook. Sage Publications Ltd., CA
98. Scripps, J., Tan, P., Esfahanian, A. (2007) Node roles and community structure in networks. In Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis (San Jose, California, August 12 - 12, 2007). WebKDD/SNA-KDD '07. ACM, New York, NY, 26-35
99. Shi, J., Malik, J. (2000) Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8):888905
100. Šíma, J., Schaeffer, S.E. (2006) On the NP-Completeness of Some Graph Cluster Measures. Proceedings of SOFSEM 2006: Theory and Practice of Computer Science, 530-537
101. Simpson, E. (2008) Clustering Tags in Enterprise and Web Folksonomies. Technical Report HPL-2008-18
102. Specia, L., Motta, E. (2007) Integrating Folksonomies with the Semantic Web. Lecture Notes In Computer Science 4519, Springer-Verlag, Berlin, Heidelberg, 624-639
103. Sun, J., Faloutsos, C., Papadimitriou, S., Yu, P. S. (2007) GraphScope: parameter-free mining of large time-evolving graphs. Proceedings of KDD '07, ACM, pp. 687-696
104. Tang, L., Liu, H. (2010) Graph Mining Applications to Social Network Analysis. In Managing and Mining Graph Data, Editors: Charu Aggarwal and Haixun Wang. Springer
105. Tsatsou, D., Papadopoulos, S., Kompatsiaris, I., Davis, P.C. (2010) Distributed Technologies for Personalized Advertisement Delivery. In Online Multimedia Advertising: Techniques and Technologies, IGI Global (accepted for publication)
106. Tyler, J. R., Wilkinson, D. M., Huberman, B. A. (2003) Email as spectroscopy: automated discovery of community structure within organizations. In Communities and Technologies, Kluwer B.V., Deventer, The Netherlands, 81-96
107. Vragović, I., Louis, E. (2006) Network community structure and loop coefficient method, Physical Review E 74, 016105
108. Wang, Y., Wu, B., Du, N. (2008) Community Evolution of Social Network: Feature, Algorithm and Model. Eprint arXiv: 0804.4356
109. Wasserman, S., Faust, K. (1994) Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge
110. Xu, X., Yuruk, N., Feng, Z., Schweiger, T. A. (2007) SCAN: A Structural Clustering Algorithm for Networks. Proceedings of KDD '07, ACM, pp. 824-833
111. Yang, B., Liu, D.-Y. (2006) Force-Based Incremental Algorithm for Mining Community Structure in Dynamic Network. Journal of Computer Science and Technology 21(3): 393-400
112. Yang, S., Wang, B., Zhao, H., Wu, B. (2009) Efficient Dense Structure Mining Using MapReduce. In Proceedings of International Conference on Data Mining Workshops, pp. 332-337
113. Ye, S., Lang, J., Wu, F. (2010) Crawling Online Social Graphs. Proceedings of 12th International Asia-Pacific Web Conference, APWeb 2010
114. Zhang, Y., Wang, J., Wang, Y., and Zhou, L. (2009) Parallel community detection on large networks with propinquity dynamics. Proceedings of KDD '09, ACM, pp. 997-1006
115. Zakharov, P. (2006) Thermodynamic approach for community discovering within the complex networks: LiveJournal study. Eprint arXiv:physics/0602063
116. Zhao, Q., Mitra, P., Chen, B. (2007) Temporal and information flow based event detection from social text streams. In Proceedings of the 22nd National Conference on Artificial intelligence (Vancouver, British Columbia, Canada, July 2007), AAAI Press, 1501-1506