Soc. Netw. Anal. Min. (2014) 4:170
DOI 10.1007/s13278-014-0170-z

ORIGINAL ARTICLE

Excavating social circles via user interests

Diana Palsetia · Md. Mostofa Ali Patwary · Ankit Agrawal · Alok Choudhary

Received: 12 June 2013 / Revised: 4 October 2013 / Accepted: 27 November 2013

Abstract The rapid evolution of modern social networks motivates the design of networks based on *users' interests*. Using popular social media such as Facebook and Twitter, we show that this new perspective can bring more meaningful information about the networks. In this paper, we model user-interest-based networks by deducing intent from social media activities such as comments and tweets of millions of users in Facebook and Twitter, respectively. These interactive contents derive networks that are dynamic in nature as the user interests can evolve due to temporal and spatial activities occurring around the user. To excavate social circles, we develop an approach that iteratively removes the influence of the communities identified in the previous steps by widely used Clauset, Newman, and Moore (CNM) community detection algorithm. Experimental results show that our approach can detect communities at a much finer scale compared to the CNM algorithm. Our user-interest-based model and community extraction methodology together can be used to identify target communities in the context of business requirements.

D. Palsetia (✉) · Md. M. A. Patwary · A. Agrawal · A. Choudhary
Electrical Engineering and Computer Science Department, Northwestern University, Evanston, IL 60208, USA
e-mail: drp925@eecs.northwestern.edu

Md. M. A. Patwary
e-mail: mpatwary@eecs.northwestern.edu

A. Agrawal
e-mail: ankitag@eecs.northwestern.edu

A. Choudhary
e-mail: choudhar@eecs.northwestern.edu

Keywords Community detection · Extraction · Analysis clustering · social network · Graph partitioning

1 Introduction

Usually complex systems are represented by network models (Strogatz 2001). There exists a wide range of such systems, for example, acquaintance and collaboration networks in sociology (Amaral et al. 2000; Newman 2001), metabolic and gene networks in biology (Fell and Wagner 2000; Jeong et al. 2000; Pons and Latapy 2004), cosmological simulation in physics (Liu et al. 2003; Skory et al. 2010), internet (Faloutsos et al. 1999), World Wide Web (Albert et al. 1999; Broder et al. 2000), social networks (Wasserman and Faust 1994), citation networks (Redner 1998; Price 1965), and food networks (Williams and Martinez 2000). Understanding and analyzing the structure of these systems has seen a surge of interest in recent years. A fundamental problem in the study of networks is community detection (Newman 2006).

In this paper, we focus on finding communities in social networks, also known as *social circles*. Due to the increasing popularity of social media sites such as Facebook[1] and Twitter[2], there is a vast amount of creation and exchange of user-generated content. In the past, experiments have been performed using traditional user networks (Clauset et al. 2004; Wakita et al. 2007). For example, in Twitter, user networks are given by *follower & following* relationships. Instead of using the static user networks, in this work we use *users' interests* to determine the social network structure. The *users' interests* are deduced from

[1] http://www.facebook.com/.
[2] http://twitter.com/.

user-generated content such as posts, comments, and likes with respect to Facebook, and tweets (retweets and mentioned tweets) with respect to Twitter. According to (Corallo 2013), on average, a Facebook user writes 25 *comments* and clicks the *like* button 9 times per month. These interests build the network connections, and the common content generators determine the strength of those connections. Using interest-based modeling also makes these networks much more dynamic in nature as user interests can evolve due to temporal (e.g., current events) or spatial (e.g., change in geographical locations) reasons. Recent work (Kashoob and Caverlee 2012; Nair and Dua 2012; Tchuente et al. 2013) has used dynamic and static social networks to propose recommender systems or personalized systems that can perform user profiling and community detection. According to Amis (2007), social network sites have as much influence on consumers as television or newspapers. More importantly, social networks are formed between users of similar interests and are peer-to-peer (Smith et al. 2005). Thus, one application of our user-interest-based social circles would be in Marketing where marketers are always looking to learn how the consumers are influenced by their environment.

Network models are often represented by graphs. The set of *nodes* or *vertices* in the graphs represent, for instance, people in social networks. The *links* or *edges* between the vertices reflect the relationships between users. These graphs are in general globally sparse, but locally dense, that is, there exists groups of vertices, called *communities*, with higher density of edges inside the groups than between the groups. These communities often bring out meaningful information about the networks.

Although there exist several community detection algorithms (discussed in Sect. 3), several algorithms are based on maximizing a metric, known as *modularity* (Fortunato 2010). However, a maximum modularity does not necessarily reflect that a network has a community structure (Fortunato 2010; Fortunato and Barthelemy 2007; Radicchi 2013). In particular, it remains true even if the communities are cliques (Fortunato and Barthelemy 2007). To facilitate the maximization of the modularity metric, these algorithms end up producing several big communities along with only few small communities.

We therefore propose an iterative approach for extracting focused social circles. In our approach, the algorithm starts with the entire network and outputs few small communities in every round. It then removes these extracted communities from the network and runs the algorithm recursively for each of the big communities. The algorithm continues until it is not possible to further divide the communities into smaller ones or until each of the communities reaches a user-desired upper bound on the community size. The proposed technique has been experimented using Facebook and Twitter data. Experimental results show that our proposed algorithm can excavate focused social circles that conform with objectives, facts, and intuitions.

Preliminary results of this work have been earlier presented in Palsetia et al. (2012). In this paper, we provide in-depth discussion of our algorithm including its complexity, and experiments pertaining to both static and dynamic datasets. The remainder of this paper is organized as follows. We describe our approach to model social networks in Sect. 2. Section 3 presents a literature review on community extraction algorithms followed by our new approach to extract communities in Sect. 4. In Sect. 5, we describe our experimental methodology, and the results. We conclude our work and propose future work in Sect. 6.

2 Data modeling

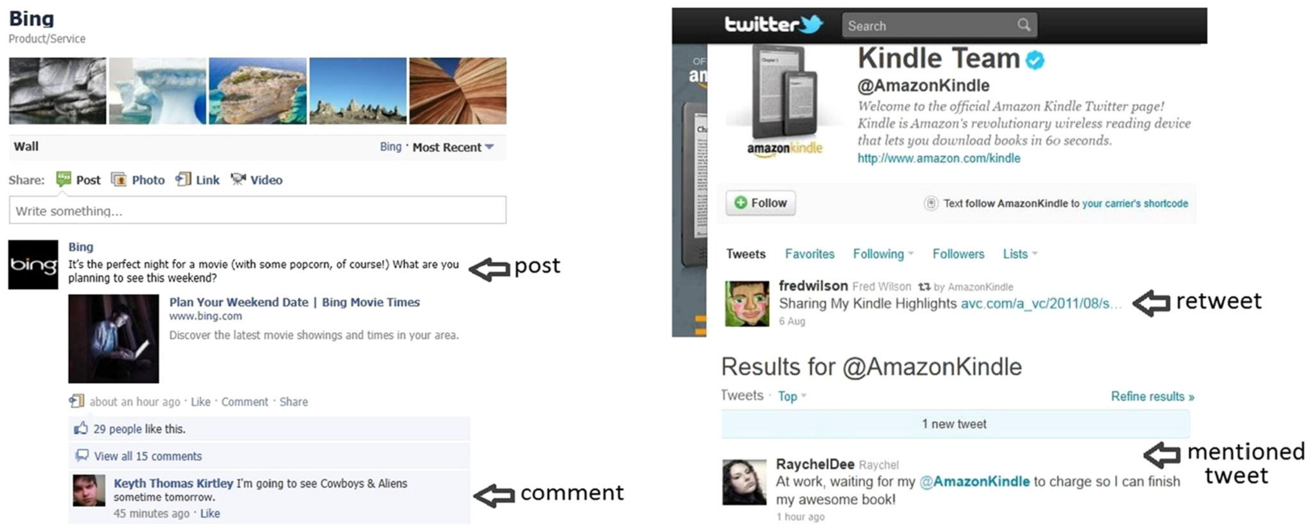
Our data is collected from two widely used social media platforms: Facebook and Twitter. Both Facebook *walls* and Twitter *profiles* are a medium for individuals, groups or businesses to post content such as messages, promotions or campaigns. These sites also provide the capability for other users to interact and engage by allowing them to reply or comment on the already posted content. To deduce *users' interests* on Facebook, we therefore consider user comments made for post on the Facebook *walls* and use them to formulate the network for our experiments. In Fig. 1a, for example, a post made by *Bing* on its wall and comments made by interested users are shown. The user comments and user information from specific *walls* are publicly available and collected using Facebook API[3].

Similarly, for Twitter the interest is deduced by *tweets*. A *tweet* is a message with up to 140 characters pertaining to a particular Twitter *profile*. We use two kinds of tweets. First, *retweet*, which is a tweet made by a Twitter *profile* that gets tweeted again by another interested user. Second, *mentioned tweet*, which is a tweet made by an interested user regarding the Twitter *profile*. In Fig. 1b, we provide an example of a Twitter *profile* named *@AmazonKindle*, retweet of *@AmazonKindle* tweet, and mentioned tweet regarding *@AmazonKindle*. The publicly available user mentioned tweets, retweets of a Twitter *profile*, and information of users who tweeted on these *profiles* are collected using Twitter API[4].

We now present our technique to generate the user-interest-based networks. From the gathered data we assimilate information regarding unique users, who have shown interest on, for example a Facebook *wall*. This is

[3] http://developers.facebook.com/.

[4] https://dev.twitter.com/docs/.



Bing
Product/Service

Wall — Bing · Most Recent

Share: Post Photo Link Video

Write something...

Bing
It's the perfect night for a movie (with some popcorn, of course!) What are you planning to see this weekend?

Plan Your Weekend Date | Bing Movie Times
www.bing.com
Discover the latest movie showings and times in your area.

about an hour ago · Like · Comment · Share

29 people like this.

View all 15 comments

Keyth Thomas Kirtley I'm going to see Cowboys & Aliens sometime tomorrow.
45 minutes ago · Like

Kindle Team
@AmazonKindle
Welcome to the official Amazon Kindle Twitter page! Kindle is Amazon's revolutionary wireless reading device that lets you download books in 60 seconds.
http://www.amazon.com/kindle

Follow — Text follow AmazonKindle to your carrier's shortcode

Tweets Favorites Following Followers Lists

fredwilson Fred Wilson by AmazonKindle
Sharing My Kindle Highlights avc.com/a_vc/2011/08/s...
6 Aug

Results for @AmazonKindle
Tweets · Top — Refine results »

1 new tweet

RaychelDee Raychel
At work, waiting for my @AmazonKindle to charge so I can finish my awesome book!
1 hour ago

(a) Facebook **(b)** Twitter

Fig. 1 User interests deduced from user-generated content

Table 1 Common User Matrix for 6 Twitter *profiles*

	h_1	h_2	h_3	h_4	h_5	h_6
h_1	30,527	13	80	4	18	84
h_2	13	1,324	30	0	6	11
h_3	80	30	6,679	18	28	87
h_4	4	0	18	358	7	20
h_5	18	6	28	7	1,961	140
h_6	84	11	87	20	140	7,951

done by extracting user identifiers from Facebook *comments* made for specific *walls*. Further, we also determine the common users between any two Facebook *walls*. We represent this data as a symmetric square matrix, M, of dimension equal to the number of *walls*. Each diagonal entry of M, say, $M[i, i]$ represents the number of unique users of *walli* and any other entries $M[i, j]$, where $i \neq j$ denote the number of common users between *walli* and *wallj*. For a *walli*, high value of $M[i, i]$ indicates the popularity of i amongst the masses. Between two *wallsi* and j, higher value of $M[i, j]$ indicates that more users are interested in both *walls*. Note that users (who generate content) are not walls (for whom the content is generated). The same procedure is carried out for Twitter *profiles*. Table 1 shows a sample matrix representation for Twitter profiles *VanCanucks* (h_1), *Versace* (h_2), *VirginAmerica* (h_3), *virginmobileus* (h_4), *Walgreens* (h_5), and *Walmart* (h_6).

Since the usual community extraction algorithms take graphs as input, we convert each such matrix to an undirected graph $G = (V, E)$, where V represents the walls or profiles and E represents edges between them. We assign an edge between two walls or profiles if the common user count is greater than zero. Also, since we want to find communities of similar interests, the edges contain weights to indicate strength of the connection. The weight between two vertices i and j, denoted by $w[i, j]$, is computed by Jaccard index or similarity coefficient (Guha et al. 2000; Leydesdorff 2008) as shown in Eq. 1.

$$w[i,j] = \frac{M[i,j]}{M[i,i] + M[j,j] - M[i,j]} \qquad (1)$$

The denominator represents the number of unique users and the numerator represents the number of common users in Eq. 1. The weight ranges between 0 and 1, with a value closer to 1 indicating that the *walls/profiles* are more similar in terms of user interests.

3 Community extraction

In this section, we provide a brief literature review of existing community detection algorithms with the widely used Clauset, Newman, and Moore (CNM) algorithm (Clauset et al. 2004).

Throughout the paper, we consider an undirected graph $G = (V, E)$ with $n = |V|$ and $m = |E|$, where V is the set of vertices and E is the set of edges. We assume that the graph is simple, meaning that there is no vertex linked to itself and there are no multiple edges between two vertices. We call $G_i = (V_i, E_i)$, a *subgraph* of G if $V_i \subseteq V$ and $E_i \subseteq E$. We use $V(G)$ and $E(G)$ to denote the vertices and edges of graph G, respectively.

The community detection problem is typically formulated as finding a partition $C = \{c_1, \ldots, c_k\}$ of G, where $\forall_i, c_i \subseteq V$ and $\forall_{i,j}, c_i \cap c_j = \emptyset$, which gives tight communities, i.e., we get many connections between the members of the community and relatively few connections among the communities. C is also known as a *clustering* of G. We use k to denote the number of resulting communities, that is, $|C| = k$.

3.1 Related works

In recent years, many new algorithms for detecting communities have been proposed, most of which belong to one of the two broad categories, *divisive* and *agglomerative*. One such divisive approach is proposed in (Girvan and Newman 2002; Newman and Girvan 2004) where the edges with largest *betweenness* (number of shortest paths passing through an edge) are removed one by one to split the graph into communities hierarchically. An alternative formulation has been proposed in Pinney and Westhead (2007), where the vertices with largest betweenness are removed instead of edges. Pinney and Westhead (2007) also presents a hybrid approach, where both vertices and edges are removed to detect overlapping communities.

Several fast agglomerative algorithms (also, known as *hierarchical* approach) have been developed in recent years (Newman 2004; Radicchi et al. 2004; Wu and Huberman 2004). Agglomerative algorithms iteratively group the vertices into communities. Different methods exist depending on the way of choosing communities to merge at each step. A greedy algorithm of this type proposed in Newman (2004) starts with n communities corresponding to the vertices of G. The algorithm then merges communities to optimize a metric called *modularity*, which is a goodness measure of a division. A division is good when there are many edges within the communities and only a few between them. This algorithm has been improved in Clauset et al. (2004, Wakita et al. (2007). Several other similarity metrics and techniques have been used in Donetti and Munoz (2004), Pons and Latapy (2004), Zhou and Lipowsky (2004).

Other than divisive and agglomerative approaches, a totally different framework for extracting communities has been recently proposed and experimented in Zhao et al. (2011). This allows some vertices not to belong to any community, whereas all the above discussed algorithms require all vertices to belong to at least one community.

The approach we introduce in this paper works on top of existing community extraction algorithms. We therefore explain our approach from the viewpoint of one such existing algorithm. Several open source software packages, for example, SNAP Stanford (Leskovec 2009), SNAP Berkeley (Madduri 2008), and iGraph (Csardi and Nepusz 2003) include the implementation of well known and widely used community extraction algorithms, for example, Girvan, Newman algorithm (Girvan and Newman 2002) and Clauset, Newman, and Moore (CNM) algorithm (Clauset et al. 2004). iGraph has few more implementations such as Donetti and Munoz (2004) and Pons and Latapy (2004). Since CNM algorithm (Clauset et al. 2004) is quite efficient and widely used, we use this in our algorithm. The source code for the CNM algorithm is available with the above mentioned packages. Next, we present a brief overview on CNM algorithm and then present our proposed algorithm in Sect. 4.

3.2 CNM algorithm

First, we formally define the modularity metric as it is the basis of the CNM algorithm. Modularity is a quantitative measure of the quality of a partition of a graph. It can be used to compare the quality of different partitions of a graph. As mentioned earlier, each community would have a high number of intra-community edges and a few inter-community edges. The formulation of modularity reflects this idea as explained next.

Let e_{ij} denotes one-half of the fraction of edges in a graph that connects vertices in community i to those in community j. Therefore, $e_{ij} + e_{ji}$ is the total fraction of such edges for communities i and j. Let e_{ii} be the fraction of edges that fall within group i. Then $\sum_i e_{ii}$ is the total fraction of edges that fall within groups and $a_i = \sum_j e_{ij}$ be the total fraction of all ends of edges that are attached to vertices in group i. Given these parameters, the modularity Q of a clustering C is defined as follows:

$$Q(C) = \sum_i (e_{ii} - a_i^2) \tag{2}$$

As can be seen in Eq. 2, to maximize modularity, the first term should be high whereas the second term should be low. This reflects the concept of community clearly. If the number of intra-community edges is no better than random, then $Q = 0$. The value of Q approaching 1, which is maximum, indicates strong community structure (Newman and Girvan 2004), although in practice, typical values fall within the range of 0.3–0.7 and higher values are rare (Fortunato and Barthelemy 2007).

CNM algorithm (Clauset et al. 2004) falls in the general category of agglomerative hierarchical clustering. The algorithm starts from a totally unclustered situation, where each vertex in a graph forms a singleton community. The algorithm then repeatedly chooses a community pair based on *modularity* and merges them into a new community. Since the number of community pairs decreases monotonically, the algorithm eventually stops when there remains no community pairs to merge. In this way, the algorithm gives a *dendrogram*, a tree that shows the order of the joins (Newman 2004). Cuts through this dendrogram at different levels give division of the graph into larger or smaller number of communities. CNM algorithm selects the best cut by looking for the maximal value of modularity as it represents the best community structure. CNM algorithm also works with weighted graphs, the only difference being that while computing modularity, it uses edge weights instead of degrees in Equation 2. More details on CNM algorithm can be found in Clauset et al. (2004) and Newman and Girvan (2004). In the rest of the paper, we use GreedyAgglomerative(GA) (Madduri 2008), to refer to the algorithm that first calls CNM algorithm for graph G, then finds the maximum modularity, and outputs a clustering C of graph G as the set of resulting communities.

4 INC algorithm

Modularity has been widely used as a metric for extracting communities in the last decade (Clauset et al. 2004; Leskovec 2009; Madduri 2008; Newman and Girvan 2004). However, according to Fortunato (2010) and Fortunato and Barthelemy (2007), maximum modularity does not necessarily mean that a graph has community structure. It is especially the case if the communities are cliques. Therefore, using modularity to extract communities results in large modules (communities), which in turn could comprise of smaller modules.

Visually analyzing the communities on several social network datasets derived from our user-interest-based model, we noticed that the social circles in general are small in size even if the dataset is large. It could also happen that if the edge densities between the social circles are strong, the existing algorithms consider them as one social circle. However, keeping them separate, we find focused communities. For example, looking at the big communities found by the GA algorithm, for Facebook and Twitter dataset, we observe that one can easily identify the focused communities related to *sports*, *politics*, *news*, and so on within large communities.

In this section, we therefore present a new approach that can further divide these big communities into small focused communities. Note that while doing so, we consider the subgraph that contains only those vertices that belong to the big community. The reason behind this is that the edges connecting a big community to other communities have already been considered while identifying the previous communities, and presently we are only interested in dividing the current big community into smaller ones. Since our approach incrementally extracts several meaningful communities in every round, we call the approach as incremental community extraction algorithm, denoted by Incre-Comm-Extraction(INC).

4.1 Our approach

Our algorithm works in a recursive fashion. At the beginning of every round, we call the GA algorithm and for each community that GA algorithm outputs, our INC algorithm either declares that as a final community or recalls our algorithm recursively for that community to divide it further. In this way, we keep dividing the input graph as long as GA algorithm can divide it into multiple communities. When GA algorithm fails to divide the input graph, our algorithm outputs that graph as a resulting community. One can also stop dividing a community when the community size reaches s, an upper bound on the community size and is an input parameter. However, this condition works as an extra feature of our algorithm if one desires to bound the communities by size s. Note that this condition does not confirm that each resulting community would at most be of size s, as there might exist communities for which GA algorithm fails to divide into multiple communities because of very high connectivity between the community members. In our approach, we use CNM, which is one of the GA algorithms, but any GA algorithm can be employed. For example, CNM algorithm can be replaced by the algorithm used in Girvan and Newman (2002).

The details of INC algorithm have been outlined in Algorithm 1. The algorithm starts with graph G and clustering $C = \emptyset$. Let, at every round of the algorithm, the running graph be denoted by G_r. The algorithm first calls GA(G_r) and gets a set of communities, C'. If $|C'| = 1$ then algorithm failed to divide the running graph G_r into more than one community. Let c_1 be that community. In this case, c_1 contains all vertices of G_r. It outputs c_1 as a resulting community by adding c_1 to C and returns from this round. If this is not the case, then it goes through each community $c_i \in C'$ and checks which of the following three cases is true. (1) If $|c_i| = 1$, then c_i is a singleton community and therefore, it adds c_i into c', which we use to denote the set of all singleton communities found in every round. c' is set to $\emptyset$ before the algorithm starts traversing the communities. At the end of the traversal, all the singleton communities are outputted by adding c' to C given that there exists at least one vertex in c'. In this way, we output all the singleton communities found in every round as one single community similar to Zhao et al. (2011). (2) The algorithm then checks whether the community size is less than or equal to s, that is, $|c_i| \leq s$. If this is the case, it adds c_i to C as a resulting community and continues to other communities. As mentioned previously, this case is only applicable when one desires to have an upper bound on the community size. (3) If none of the above two cases are true, it means that, it might be possible to further divide the subgraph G_i, containing the vertices in c_i, $V(c_i)$ and the edges in c_i, $E(c_i)$. We therefore call INC algorithm with parameter G_i, which becomes the running graph G_r in the following round. At the end of the algorithm, we thus get all the resulting k communities in C. The flow of the algorithm can be represented as a dendrogram, where each internal node represents the running subgraph G_r and each leaf node represents a community c_i.

Algorithm 1: Incre-Comm-Extraction(G_r): Template for Incremental Community Extraction (INC) algorithm.

```
Input: Graph $G = (V, E)$. $s$ denotes the maximum
       size of a community used as an optional
       feature of the algorithm
Output: Communities $C = \{c_1, c_2, \ldots, c_k\}$, where
        $|C| = k$, $\forall_i, c_i$ is a resulting community, and
        $k$ denotes the number of such communities.
1  $C' \leftarrow$ GreedyAgglomerative($G_r$)
2  if $|C'| = 1$ then
3      Let $c_1$ be the only community.
4      $C \leftarrow C \cup c_1$
5      return
6  $c' \leftarrow \emptyset$
7  for each community $c_i \in C'$ do
8      if $|c_i| = 1$ then
9          $c' \leftarrow c' \cup c_i$
10     else if $|c_i| \leq s$ then
11         $C \leftarrow C \cup c_i$
12     else
13         $G_i \leftarrow G(V(c_i), E(c_i))$
14         Incre-Comm-Extraction($G_i$)
15 if $|c'| \neq 0$ then
16     $C \leftarrow C \cup c'$
```

Note that the idea of INC algorithm is similar to Zhao et al. (2011) (although they use a much different approach called *Tabu Search*). However, the INC algorithm extracts several small communities at every round instead of extracting a single community per round as in Zhao et al. (2011). Moreover, while extracting communities from a big community, INC algorithm disregards the connections of a big community with the outside world as they have already been considered in earlier rounds. Another technique, called *HQcut*, has been presented in Ruan and Zhang (2008), which uses spectral partitioning in their approach. HQcut is able to detect communities at a much finer scale by recursively applying the algorithm on sub-networks that are eligible for further partitioning. The eligibility is based on (1) minimum threshold value of modularity and (2) estimating the statistical significance of the modularity using a Monte–Carlo method. Lastly, although our algorithm is hierarchical in nature, this is not to be confused with (Guha et al. 2000), which uses similarity metric as distance metric and then applies agglomerative hierarchical clustering algorithm to find clusters or communities.

4.2 Complexity analysis

We now analyze the running time of the INC algorithm. As shown in Clauset et al. (2004), the running time of the GA algorithm is $O(md\log n)$ for general graphs (d is the depth of the dendrogram describing the community structure), whereas for sparse graphs, the running time is $O(n\log^2 n)$. Let the complexity of the Incre-Comm-Extraction algorithm be denoted by $T(n)$.

In the worst case, the communities found at every round may not be balanced. Since the running time of GA algorithm is $n\log^2 n$ for sparse graph, the recurrence becomes

$$T(n) = n\log^2 n + T(n-1)$$

The calls will be of the type $T(n-1)$, $T(n-2)$, $T(n-3)$, …, $T(2)$, $T(1)$. Therefore, the running time for INC algorithm for sparse graphs is $O(n^2 \log^2 n)$. Using similar analysis, one can show that for general graphs, $T(n) = O(mnd\log n)$ for INC algorithm assuming the running time of GA algorithm is $md\log n$ (Clauset et al. 2004). In the worst case scenario, the INC algorithm should call the GA algorithm n times. However, our experiments show that this number is much smaller than n.

In the case of balanced communities (most likely case), say that atmost p communities are found by the algorithm at every round and each of these communities has n/p members. We can then bound recurrence to $(\log n/\log p) * n * \log^2 n$ for sparse graphs. Without p and assuming $p \geq 2$ we get: $T(n) = O(n \log^3 n)$. Similarly, for general graphs, the recurrence then becomes $T(n) = O(mnd\log n)$.

4.3 Cluster quality metric

As mentioned previously, modularity might not reflect the right community structure, we therefore use *modularity density*, introduced in Zhang et al. (2009), as a metric to compare the quality of the extracted communities in our experiments. For a given subgraph $G_i(V_i, E_i)$, let l_i and $\bar{l}_i$ denote the number of inner (both end in G_i) and outer (only one end in G_i) edges respectively. Assuming $n_i = |V_i|$, the average inner degree of G_i is $2l_i/n_i$ and the average outer degree of G_i is $\bar{l}_i/n_i$. Then the modularity density, D of a partition $C = \{c_1, \ldots, c_k\}$ of a graph G is computed as the sum of the average inner degree minus the average outer degree. This is shown in Equation 3. The larger the value of D, the more accurate the partition is.

$$D = \sum_{i=1}^{|C|} \frac{2l_i - \bar{l_i}}{n_i} \tag{3}$$

5 Experiments and results

In this section, we analyze the communities extracted using the proposed INC algorithm.

Table 3 Structural properties of the social networks

	fb_dyn	*tw_dyn*	*tw_stat*
Edges	965,605	33,418	3,703
Maximum degree	1,937	303	125
Minimum degree	0	0	0
Average degree	965	197	22
Singleton vertices	33	20	27
Connected components	34	21	29

Table 2 Matrix structural properties

	Facebook	Twitter
Max (*uu*)	766,700	173,100
Min (*uu*)	0	0
Avg (*uu*)	14,070	6,946
Max (*cu*)	30,443	11,907
Min (*cu*)	0	0
Avg (*cu*)	10	46
Total (*uu*)	22,795,352	2,215,581

5.1 Experimental setup

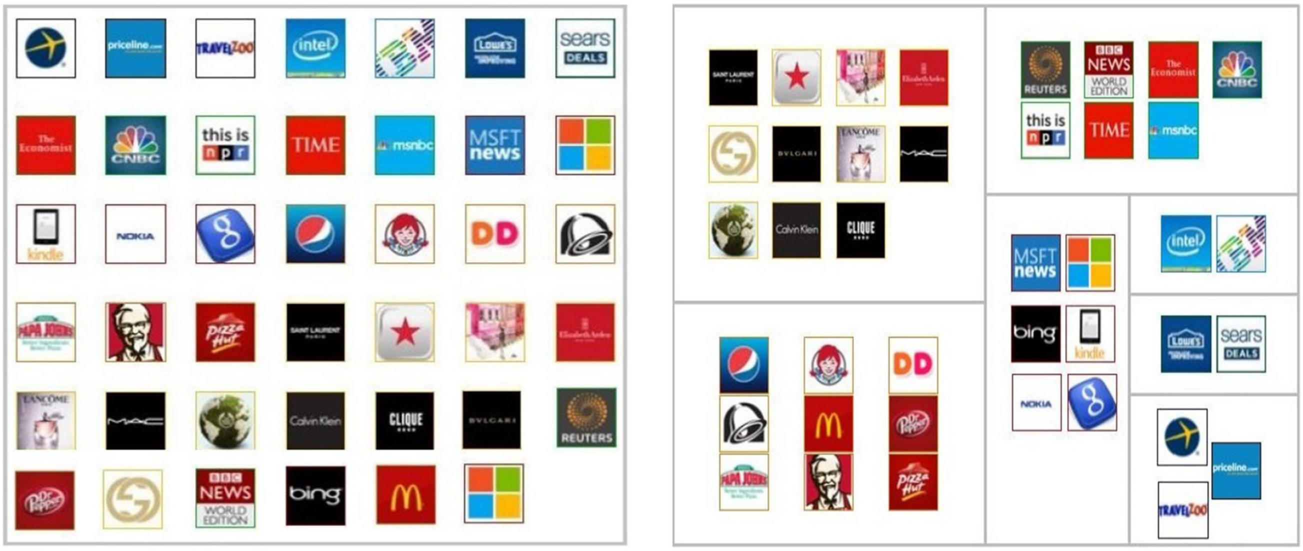
In our experiments, we consider 2,000 Facebook *walls* and 339 Twitter *profiles*. To serve as ground truth for our community extraction algorithm, we chose known walls and profiles from interests such as *sports*, *news*, *politics*, *business*, *travel*, and *entertainment*. We therefore have 2,000 × 2,000 matrix for Facebook and 339 × 339 matrix for Twitter. Overall we have 22, 795, 352 unique Facebook users and 2, 215, 581 unique Twitter users. The data (comments and tweets) are collected over period of 2011 and 2012. Table 2 provides structural information of the Facebook and Twitter matrices. For the matrices, we denote the number of common users and unique users by *cu* and *uu*, respectively.

Twitter provides *follower* and *following* networks through their API; we therefore are able to generate static network for the same 339 Twitter *profiles*. The structural properties of the Twitter and Facebook networks are given in Table 3. The Twitter networks are denoted by *tw_stat* (static) and *tw_dyn* (dynamic). The Facebook network is denoted by *fb_dyn* (dynamic). Note that the user's friend and follower networks in Facebook are not available publicly; we therefore do not have static network for Facebook in our experiments.

For the experiments, we have used a server with Intel(R) Xeon(R) E7540 processor running at 2 GHz with total memory size of 256 GB. The operating system is GNU/Linux. All our algorithms and the SNAP-Berkeley package (Madduri 2008) that we used in our experiments were implemented in C and compiled with GCC version 4.4.5 using the -O3 optimization flag.

Table 4 Frequencies of community and clique size in Twitter networks

	tw_dyn			*tw_stat*				*tw_dyn*			*tw_stat*		
Size	*inc*	*cnm*	*cliq*	*inc*	*cnm*	*cliq*	Size	*inc*	*cnm*	*cliq*	*inc*	*cnm*	*cliq*
1	20	20	50	27	27	101	15	0	0	1	–	–	–
2	16	1	12	23	2	37	21	–	–	–	1	0	0
3	13	2	1	29	0	13	26	–	–	–	0	0	1
4	14	0	3	19	1	5	31	0	0	1	–	–	–
5	11	0	2	7	0	5	32	0	0	1	–	–	–
6	4	0	0	3	0	3	37	–	–	–	0	1	0
7	5	1	1	3	0	2	71	–	–	–	0	1	0
8	2	0	1	1	0	0	73	0	1	0	–	–	–
9	2	0	0	–	–	–	103	0	1	0	–	–	–
10	2	0	0	–	–	–	128	0	1	0	–	–	–
11	1	0	0	0	0	1	147	0	0	1	–	–	–
12	–	–	–	0	1	1	184	–	–	–	0	1	0
13	1	0	0	–	–	–	–	–	–	–	–	–	–



(a) *Single community by CNM algorithm* **(b)** *Communities by INC algorithm*

Fig. 2 Community Extraction using CNM and INC

5.2 Results and discussion

Table 4 shows the number of communities found by the INC and the CNM algorithm (denoted by *inc* and *cnm*, respectively) for Twitter networks. The table also shows the number of disjoint cliques (denoted by *cliq*). As the clique size increases, frequency of the number of cliques decreases. Most of the cliques have size less than 5. The table shows that the CNM algorithm generates large communities and therefore is not capable to identify these small cliques. This is because CNM algorithm merges most of the cliques to form a large community (sizes of 73, 103, and 128 for *tw_dyn* and sizes of 71 and 184 for *tw_stat*) to maximize modularity. On the other hand, our INC algorithm has been able to successfully extract the small size cliques as the frequencies of the small size communities are higher than large size communities. We observed similar results for Facebook network.

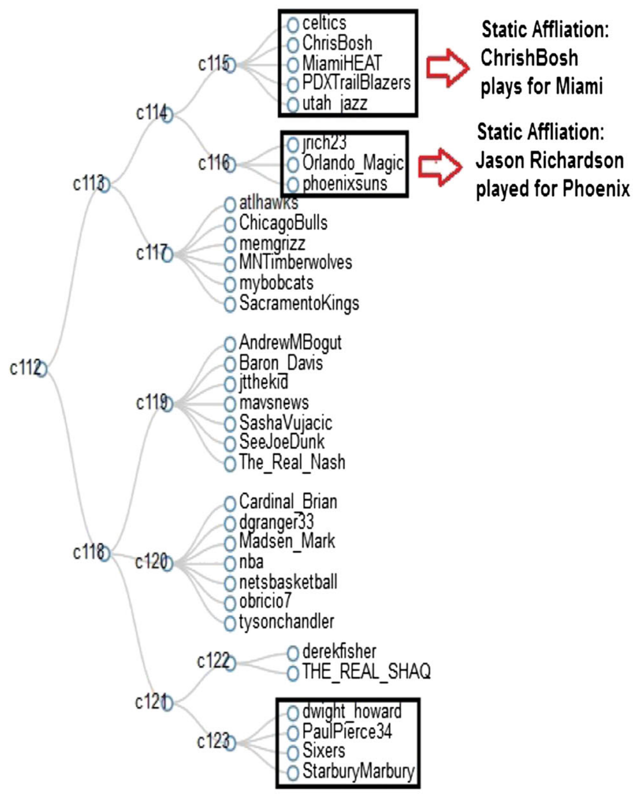
Applying CNM algorithm on the Twitter dataset resulted in three large communities. Figure 2a shows one of the three communities. As can be seen, the large community consists of several small communities from different interest groups. Figure 2b is the result of applying our INC algorithm. For example, the box comprising of *expedia*, *PricelineNegotiator*. and *Travelzoo* represents the *Travel* interest group. Other interest groups include *technology*, *fast food*, *beauty and cosmetics*, and *news agencies*. Thus the goal of our approach is to automatically discover small vut meaningful social circles.

Next, we showcase the extracted communities using our algorithm. Usually the communities found by community extraction algorithms are represented as dendrograms (Clauset et al. 2004; Girvan and Newman 2002; Newman 2004). Since the dendrograms are quite big in our case, we present partial dendrograms both for Facebook (Fig. 4) and Twitter (Fig. 3) drawn using D3, a javascript-based application[5]. Each internal node in the dendrograms represents a community whereas each leaf node is a *wall/profile*. Note that we do not experiment with the optional feature s to generate these dendrograms, i.e., the algorithm continues until it is possible to divide sub-network. The full dendrograms of all the above mentioned graphs can be found at our website[6]. Since we used known 2,000 Facebook *walls* and 339 Twitter *profiles*, we are able to verify the category of each wall and label the *wall/profile* and other affiliations in the group as well. Note that even though we only look at small number of Facebook *walls* or Twitter *profiles*, the number of users extracted to form the user-interest-based network is 22,795,352 for Facebook and 2,215,581 for Twitter.

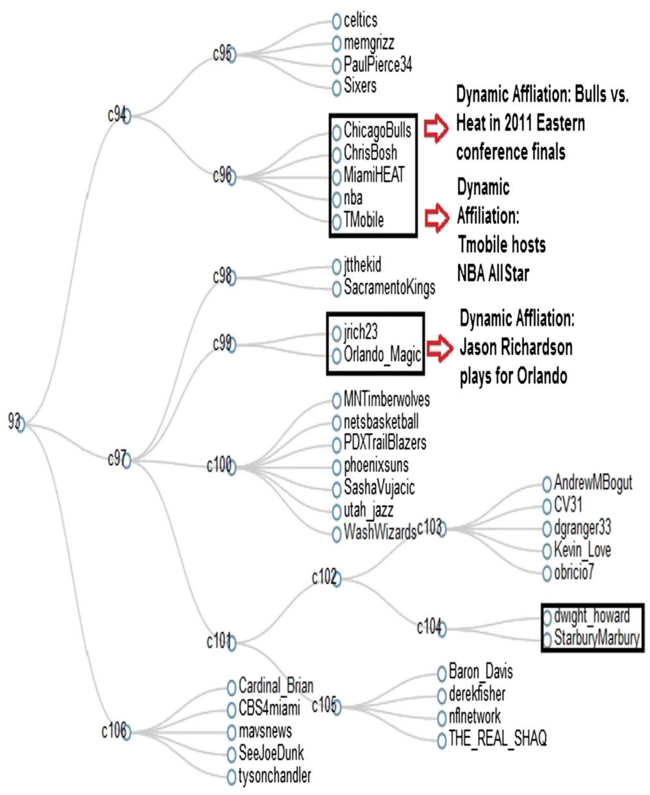
Our approach has been able to extract focused communities for both Twitter and Facebook networks. We first showcase the social circles excavated from both static and dynamic (as described in Sect. 2) Twitter networks. In Fig. 3, partial dendrograms display communities from categories *NBA Basketball* and *Republican Politicians*. For both categories our user-interest-based approach is able to closely model the explicit static network and finds more affiliations based on the user interests. For example community c_{123} in Fig. 3a and c_{104} in Fig. 3b captures relationship between two basketball players Dwight Howard

[5] http://mbostock.github.com/d3/.

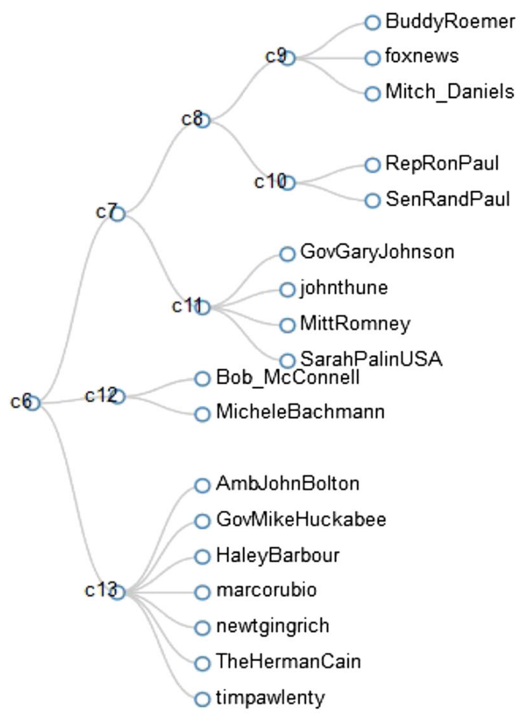
[6] http://pulse.eecs.northwestern.edu/~drp925/inc/graph.php.



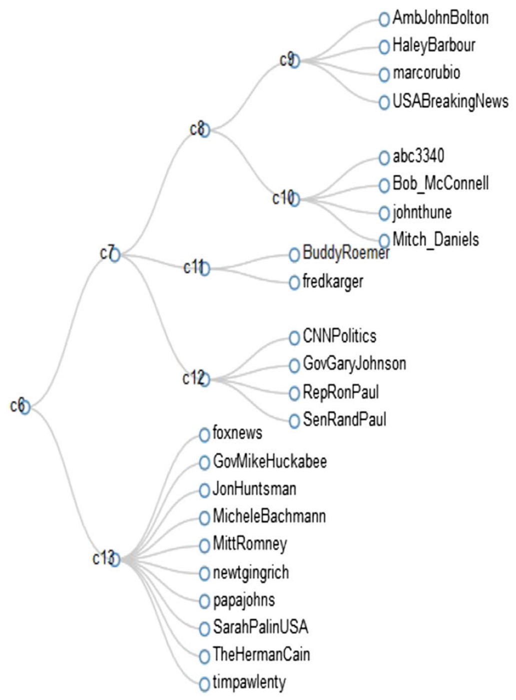
(a) *NBA Basketball - Static*



(b) *NBA Basketball - Dynamic*



(c) *Republican Politicians - Static*



(d) *Republican Politicians - Dynamic*

Fig. 3 Partial dendrograms showing communities in Twitter

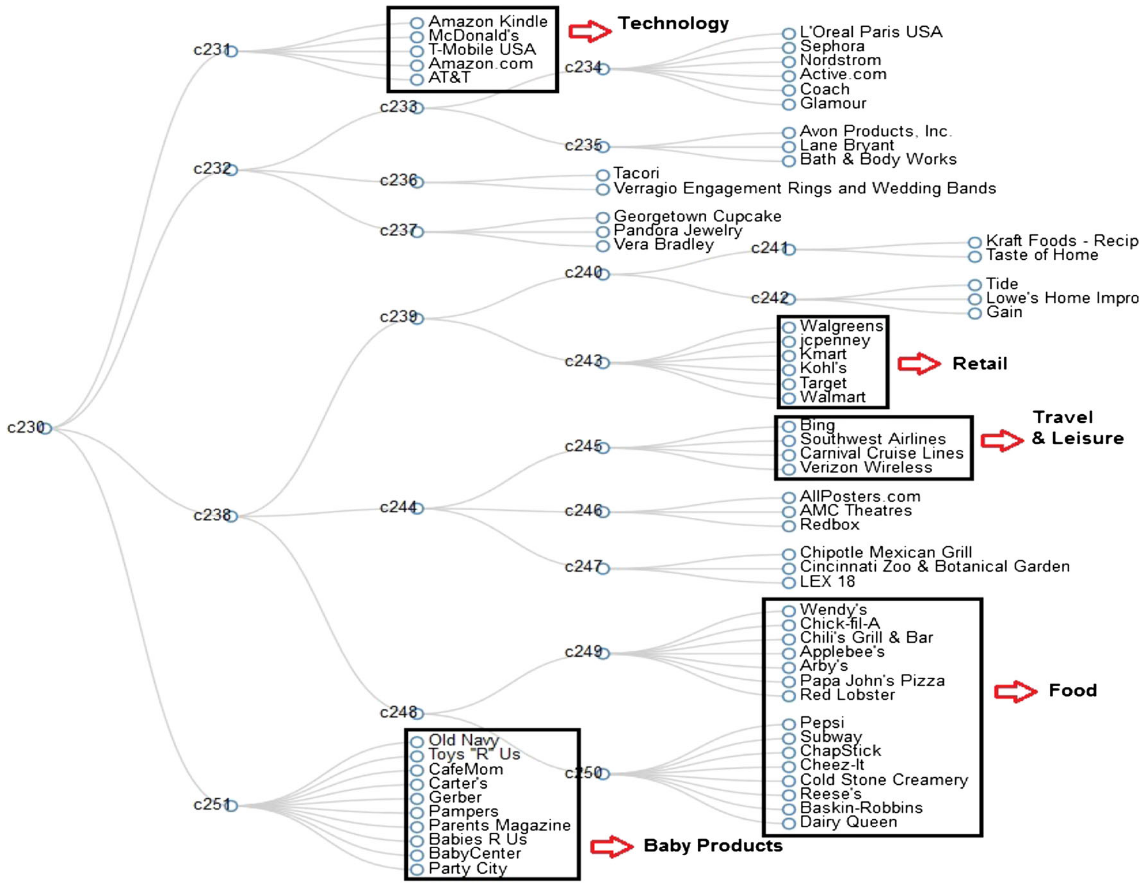


Fig. 4 Partial dendrogram showing communities in Facebook

(*dwight_howard*) and Stephon Marbury (*StarburyMarbury*). The community c_{104} does not caputure all affliations comprised in c_{123} as user interest on these affliations may be little to none. Also, in Fig. 3b, TMobile is captured in community c_{96} because TMobile hosted many events at the 2011 NBA All-Star Game[7]. This is a very interesting result because dynamic networks are constantly changing, and these results demonstrate that we can capture affiliations which are temporal in nature. Thus strengthening the case for viewing dynamic networks compared to static networks. Figure 3a, b also provides some more static and dynamic associations. In the case of *Republican Politicians*, couple of news agencies (e.g., foxnews, CNNPolitics, USABreakingNews) actively following the politicians also get captured (see Fig. 3d). Also the dynamic network is able to capture more affiliations that may not be labeled similar but indirectly affiliated, if looked at closely. Therefore, we believe this study will help businesses to identify target communities for marketing purposes. For example, in social ads targeting, excavated social circles would enable marketers to reach customer lookalikes, their networks, thereby providing a way to acquire, retain and grow customers, fans, and followers.

In Fig. 4, we look at a partial branch of the dendrogram of Facebook network represented by node c_{230}. Our approach was successfully able to extract several social circles that belong to categories such as *Technology*(c_{231}), *Retail*(c_{243}), *Travel & Leisure*(c_{244}), *Baby Products*(c_{251}), and *Food* (c_{248}). We got many more interesting focused communities, which can be found at our project website[6]. Note that most of these focused communities found by our INC algorithm belonged to one large community using CNM algorithm, and their identification has been made possible using the proposed work in this paper.

To compare the quality between CNM and INC algorithm we use *modularity density* (described in Sect. 4.3). Table 5 shows the modularity (Q) and modularity density (*mod_den*) for the Twitter and Facebook networks. Higher values of Q and *mod_den* mean better community structure. As can be seen, Q is always higher for the CNM algorithm compared to the INC algorithm, whereas *mod_den*, which is a better quality metric, is always higher for the INC algorithm than the CNM algorithm.

[7] http://newsroom.t-mobile.com/articles/t-mobile-nba-all-star-wade-barkley-basketball.

Table 5 Comparison between CNM and INC algorithm

	CNM		INC	
	Q	*mod_den*	*Q*	*mod_den*
fb_dyn	0.11	2,428.47	0.00	2,622.88
tw_dyn	0.10	443.08	0.01	486.44
tw_stat	0.31	136.04	0.10	505.53

We have further compared the HQcut[8] algorithm (Ruan and Zhang 2008) to the CNM and INC algorithms. Since the source code is limited to unweighted graphs, we were only able to produce results for the *tw_stat* dataset. For this dataset, HQcut has *Q* value of 0.47, which is higher than both CNM (0.31) and INC (0.10). The *mod_den* value for HQcut is 178.16, which is better than CNM (136.04) but is still lower compared to INC (505.53). Since higher value of modularity density means better the quality of clustering, therefore, INC performs better than HQcut. Note that the value of *Q* and *mod_den* may vary for the same dataset as HQcut estimates the statistical significance of modularity using Monte–Carlo method to further divide the sub-network recursively. Finally, even though HQcut is recursive in nature, it does not give hierarchical information about the sub-networks. As modularity density delivers better results than modularity (Fortunato 2010), INC algorithm recovers natural communities from the social networks compared to the CNM and HQcut algorithms.

Finally, we compare the timing results between the GA and INC algorithms. In the case of Facebook dataset, the INC algorithm takes 4 times longer to run. With respect to Twitter datasets, the INC algorithm takes 3.7 times for dynamic network and 18.3 times for the static network. As expected, the INC algorithm takes longer time because of its iterative nature, with each iteration running CNM algorithm on different subgraphs, but the tradeoff is the ability to discover more focused and meaningful communities, which can provide actionable insights for various business and marketing applications.

6 Conclusion and future work

In this paper, we model social networks from *user-generated* content. Instead of using the traditional user networks of Facebook and Twitter, we deduce user-interest-based networks using posts, comments, and tweets of millions of users. We show that this model closely captures relations found in static networks and can also finds affiliations that are constantly evolving either due to temporal or spatial activities. Further, we develop a new approach for mining communities to understand and analyze the structure of social networks.

To overcome the limitations of the widely used modularity-based algorithm (CNM), our approach incrementally extracts communities disregarding the influence of the communities identified in the previous steps. Our user-interest-based model and community extraction algorithm together can be used to identify social circles in the context of business requirements.

In future, we intend to experiment with time-based user-interest modeling to study effects of temporal events on the community structure. Further we are interested in developing clique-based community extraction algorithm that allows single user to belong to multiple communities and parallelize these algorithms for big data.

Acknowledgments We thank William Hendrix for helpful discussions and comments on the preliminary drafts of the paper. We thank Christopher Moran for helping us plotting the dendrograms and Kathy Lee, Kunpeng Zhang, Yves Xie, and Daniel Honbo for collecting testdata from Facebook and Twitter. This work is supported in part by the following grants: NSF awards CCF-0833131, CNS-0830927, IIS-0905205, CCF-0938000, CCF-1029166, ACI-1144061, and IIS-1343639; DOE awards DE-FG02-08ER 25848, DE-SC0001283, DE-SC0005309, DESC0005340, and DESC0007456; AFOSR award FA9550-12-1-0458.

References

Albert R, Jeong H, Barabasi A (1999) Diameter of the world-wide web. Nature 401:130–131

Amaral LAN, Scala A, Barthelemy M, Stanley HE (2000) Classes of small-world networks. PNAS 97(21):11149–11152

Amis R (2007) You can't ignore social media: How to measure internet efforts to your organisation's best advantage. Tactics, p 10

Broder A, Kumar R, Maghoul F, Raghavan P, Rajagopalan S, Stata R, Tomkins A, Wiener J (2000) Graph structure in the web. Comput Netw 33:309–320

Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70(6):066,111

Corallo M (2013) The average facebook user. http://edudemic.com/wp-content/uploads/2013/02/facebook-average-user.jpg

Csardi G, Nepusz T (2003) The igraph library. http://igraph.sourceforge.net

Donetti L, Munoz MA (2004) Detecting network communities: a new systematic and efficient algorithm. J Stat Mech Theory Exp 2004(10):10012

de Price DJS (1965) Networks of scientific papers. Science 149:510–515

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

Fell DA, Wagner A (2000) The small world of metabolism. Nature Biotechnol18:1121–1122

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174

Fortunato S, Barthelemy M (2007) Resolution limit in community detection. Proc Natl Acad Sci 104(1):36

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. PNAS 99(12):7821–7826

[8] http://cs.utsa.edu/~jruan/qcut.tar.

Guha S, Rastogi R, Shim K (2000) Rock: a robust clustering algorithm for categorical attributes. Inf Syst 25(5): 345–366

Jeong H, Tombor B, Albert R, Oltvai ZN, Barabasi AL (2000) The large-scale organization of metabolic networks. Nature 407: 651–654

Kashoob S, Caverlee J (2012) Temporal dynamics of communities in social bookmarking systems. Soc Netw Anal Min 2(4): 387–404

Leskovec J (2009) Stanford network analysis project. http://snap.stanford.edu

Leydesdorff L (2008) On the normalization and visualization of author co-citation data: Salton's cosine versus the jaccard index. J Am Soc Inf Sci Technol 59(1):77–85

Liu Y, Liao W, Choudhary A (2003) Design and evaluation of a parallel hop clustering algorithm for cosmological simulation. In: International Parallel and Distributed Processing Symposium, p 82a

Madduri K (2008) Snap: small-world network analysis and partitioning. http://snap-graph.sourceforge.net

Nair V, Dua S (2012) Folksonomy-based ad hoc community detection in online social networks. Soc Netw Anal Min 2(4):305–328

Newman MEJ (2001) The structure of scientific collaboration networks. PNAS 98(2):404–409

Newman MEJ (2004) Fast algorithm for detecting community structure in networks. Phys Rev E 69(6):066133

Newman MEJ (2006) Modularity and community structure in networks. PNAS 103(23):8577–8582

Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113

Palsetia D, Patwary MMA, Zhang K, Lee K, Moran C, Xie Y, Honbo D, Agrawal A, Liao WK, Choudhary A (2012) User-interest based community extraction in social networks. In: Proceedings of the KDD workshop on social network mining and analysis (SNAKDD), pp 1–4

Pinney J, Westhead D (2007) Betweenness-based decomposition methods for social and biological networks. Interdiscipl Stat Bioinf pp 87–90

Pons P, Latapy M (2004) Computing communities in large networks using random walks. J Graph Algorithms Appl 10:284–293

Radicchi F (2013) Detectability of communities in networks. URL:http://arxiv.org/abs/1306.1102

Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D (2004) Defining and identifying communities in networks. PNAS 101:2568–2663

Redner S (1998) How popular is your paper? an empirical study of the citation distribution. Eur Phys J B 4:131–134

Ruan J, Zhang W (2008) Identifying network communities with a high resolution. Phys Rev E 77(1):016104

Skory S, Turk MJ, Norman ML, Coil AL (2010) Parallel hop: A scalable halo finder for massive cosmological data sets. Astrophys J Suppl Ser 191(1):43

Smith D, Menon S, Sivakumar K (2005) Online peer and editorial recommendations, trust, and choice in virtual markets. J Interact Mark 19(3):15–37. doi:10.1002/dir.20041

Strogatz SH (2001) Exploring complex networks. Nature 410: 268–276

Tchuente D, Canut CMF, Jessel N, Péninou A, Sèdes F (2013) A community-based algorithm for deriving users' profiles from egocentrics networks: experiment on facebook and dblp. Soc Netw Anal Min 3(3):667–683

Wakita, K, Tsurumi T (2007) Finding community structure in mega-scale social networks:[extended abstract]. In: Proceedings of the 16th international conference on World Wide Web. ACM, New York, pp 1275–1276

Wasserman S, Faust K (1994) Social network analysis. Cambridge University Press, Cambridge

Williams RJ, Martinez ND (2000) Simple rules yield complex food webs. Nature 404:180–183

Wu F, Huberman BA (2004) Finding communities in linear time: a physics approach. Eur Phys J B 38:331–338

Zhang S, Ning X, Ding C (2009) Maximizing modularity density for exploring modular organization of protein interaction networks. In: Third international symposium on optimization and systems biology, pp 361–370

Zhao Y, Levina E, Zhu J (2011) Community extraction for social networks. PNAS 108(18):7321–7326

Zhou H, Lipowsky R (2004) Network brownian motion: a new method to measure vertex–vertex proximity and to identify communities and subcommunities. Comput Sci ICCS, pp 1062–1069