User-Interest based Community Extraction in Social Networks

Diana Palsetia[†,‡], Md. Mostofa Ali Patwary[†,‡], Kunpeng Zhang, Kathy Lee, Christopher Moran, Yves Xie, Daniel Honbo, Ankit Agrawal, Wei-keng Liao, Alok Choudhary
Dept. of Electrical Engineering and Computer Science, Northwestern University
Evanston, IL 60208, USA
[†]Authors contributed equally
[‡]Corresponding authors: {drp925,mpatwary}@eecs.northwestern.edu

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 generate 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. This interactive content derives networks that are dynamic in nature as the user interests can evolve due to temporal and spatial activities occurring around the user. To understand and analyze these networks, we develop a new approach for mining communities to overcome the limitations of the widely used Clauset, Newman, and Moore (CNM) community detection algorithm. The key feature of the proposed approach is that the communities are extracted incrementally by removing the influence of the communities identified in the previous steps. Experimental results show that our approach can find many focused communities of similar interests compared to the large communities found by 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.

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Data mining; G.2.2 [**Graph Theory**]: Graph algorithms

General Terms

Algorithms, Experimentation, Measurement

Keywords

Community detection, extraction, and analysis, clustering, social network, graph partitioning.

The 6th SNA-KDD Workshop '12, August 12, 2012, Beijing, China

1. INTRODUCTION

Usually, complex systems are represented by network models to the scientific community [14]. There exists a wide range of such systems, for example, acquaintance and collaboration networks in sociology [1], and social networks [15]. Understanding and analyzing the structure of these systems has caused a surge of interest in recent years. A fundamental problem in the study of networks is community detection [11]. In this paper we focus on finding communities in social networks. Due to the increasing popularity of social media sites such as Facebook and Twitter, there is vast amount of creation and exchange of user-generated content.

In the past, experiments have been performed using traditional user networks [2]. For example, in Twitter, user networks are given by *follower & following* relationship. Instead of using the static user networks, we used *users' interests* to determine the social network. The *users' interests* are deduced from user-generated contents such as posts, comments, and likes with respect to Facebook and tweets (retweets and mentioned tweets) with respect to Twitter. These interests build the network connections (often represented by graphs) and the common content generators determine the strength of those connections. Using interest based modelling also makes these networks dynamic in nature as user interests can evolve due to temporal (e.g. current events) or spatial (e.g. change in geographical locations) reasons. Our model for generating networks with millions of users is discussed in Section 2. Using these networks, we analyze the communities formed. These communities could be important to different interest groups to identify target users for marketing purposes.

Although there exist several community detection algorithms, (discussed in Section 3), the widely used algorithms are based on optimizing a metric, known as *modularity* [3]. According to [3, 4], maximum modularity does not necessarily reflect that a network has community structure. In particular, it remains true if the communities are cliques. The inherent nature of these algorithms is that they extract several large communities along with only few small communities to maximize the modularity metric. We therefore propose an iterative approach for extracting focused communities. Note that by *focused* we mean having the same social interest. In our approach, the algorithm starts with the whole network and extracts few small communities at every round. It then removes these extracted communities

from the whole 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 reasonable size. The proposed technique has been experimented using datasets obtained from existing social networking services, for example, Facebook and Twitter. Experimental results show that this algorithm can extract focused communities that conform with both objective facts and intuitions.

The remainder of this paper is organized as follows. In Section 2 we describe our approach to model social networks. In Section 3, we present a brief literature review on community extraction algorithms and our new approach to extract communities. In Section 4, we describe our experimental methodology, and the results. We conclude our work and propose future work in Section 5.

2. DATA MODELLING

Our data is collected from two widely used social media platforms: Facebook and Twitter. *Users' interests* are drawn from Facebook *walls* and Twitter user *profiles*. Both Facebook *walls* and Twitter *profiles* are a medium for individuals, groups or businesses to post content such messages, promotions or campaigns. These sites also have the capability for other users to interact and engage by allowing users to reply or comment on the already posted content. This interactive content reflects the users' *interests* for Facebook *walls* or Twitter *profiles*. 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. The user comments, and user information from specific *walls* is publicly available and collected using Facebook API[1]. 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*. The publicly available user tweets of a Twitter *profile*, and information of users who tweeted on these *profiles* is collected using Twitter API[2]. In the experiments, the data collected up to June, 2011 is used.

From the gathered data we assimilate information regarding unique users, who have shown interests on Facebook and Twitter. This is done by extracting user identifiers from Facebook *comments* and Twitter *tweets* made for specific *walls* and *profiles*. Further, we also determine the common users between any two Facebook *walls* or Twitter *profiles*. We represent this data as a symmetric square matrix, M, of dimension equal to the number of *walls/profiles*. Each diagonal entry of M, say, $M[i,i]$ represents the number of unique users of *wall/profile* i and any other entries $M[i,j]$, where $i \neq j$ denote the number of common users between *wall/profile* i and *wall/profile* j. For a *wall/profile* i, high value of $M[i,i]$ indicates the popularity of i amongst the masses. Between two *walls/profiles* i and j, higher value of $M[i,j]$ indicates more users are interested in both *walls/profiles*.

In our experiment, 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 2000×2000 matrix for Facebook and 339×339 matrix for Twitter. Table 1 provides structural information on the Facebook and Twitter matrices. For the matrices, we denote the number of common users and unique users by cu and uu, respectively.

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

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

Table 1: Matrix structural properties

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

Since the usual community extraction algorithms take graphs as input, we convert each of these matrices 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 determined by Jaccard index (similarity coefficient) [8] as shown in Equation 1.

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

The denominator represents the number of unique users and the numerator represents the number of common users in Equation 1. The weight value is in the range 0 to 1. Therefore, value closer to 1 indicates that the *walls/profiles* are more similar.

Twitter provides follower-following relationship through their API, we therefore are able to generate static network for the same 339 Twitter *profiles*. The structural properties of the Twitter networks and Facebook network are given in Table 2. 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 networks in Facebook are not available publicly, we therefore don't have static network for our experiments.

Table 2: Structural properties of the social networks

	fb_dyn	*tw_dyn*	*tw_stat*
Edges	965,605	33,418	3703
Maximum Degree	1,937	303	125
Average Degree	965	197	22
Singleton vertices	33	20	27
Connected components	34	21	29

3. COMMUNITY EXTRACTION

The community detection problem is typically formulated as finding a partition $C = \{c_1, \ldots, c_k\}$ of a simple graph $G = (V, E)$, where $\forall_i, c_i \subseteq V$ and $\forall_{i,j}, c_i \cap c_j = \emptyset$, which gives tight or meaningful communities in some suitable sense. C is also known as a *clustering* of G. We use k to denote the number of resulting communities, that is, $|C| = k$.

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 [12] 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. Several fast agglomerative algorithms (also, known as *hierarchical* approach) have been developed in recent years [10, 13, 16]. 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 [10] starts with n communities corresponding to the vertices of G. The algorithm then merges communities in order to optimize a function called *modularity*, which is a goodness measure of a division. A division is good when there are many edges within communities and only a few between them. This algorithm has been improved in [2].

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 [7], and SNAP Berkeley [9] include the implementation of the well known and widely used community extraction algorithms, for example, Girvan, Newman algorithm [5] and Clauset, Newman, and Moore (CNM) algorithm [2]. Since CNM algorithm [2] is quite efficient and widely used, we use this in our algorithm. We now present a brief overview on CNM algorithm and then present our proposed algorithm in Section 3.2.

3.1 CNM Algorithm

We first define the modularity metric formally as this is the basis of the CNM algorithm. Modularity is a quantitative measure of the quality of a partition of a graph. This can be used to compare the quality of different clusterings of the same graph. The formulation of modularity reflects the idea of higher number of intra-community edges compared to inter-community edges as explained subsequently. 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. Therefore, the modularity Q of a clustering C is defined as:

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

As can be seen in Equation 2, to maximize modularity, the first term should be high whereas the second term should be low. This reflects the concept of community clearly. The value of Q approaching 1 indicates strong community structure [12]. 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 is while computing modularity, it uses edge weights instead of degrees in Equation 2. More details on CNM algorithm can be found in [2, 12]. In the rest of the paper by GreedyAgglomerative(GA) algorithm [9], we mean an algorithm that first calls CNM algorithm for graph G, then finds the maximum modularity, and outputs the clustering C of graph G as the set of resulting communities.

3.2 Our Approach

Modularity has been widely used as a metric in extracting communities in the last decade [2, 7, 9, 12]. However, according to [4], maximum modularity does not necessarily mean that a graph has community structure. In particular, it remains true if the communities are cliques. Therefore, using modularity to extract communities results in large modules (communities), which in turn could be comprised of smaller modules.

Visually analyzing the communities on several social network datasets derived from our user-interest based model, we noticed that the communities in general are small in size even if the dataset is large. It could also happen that the edge densities between communities are high and deserve merging between them, although keeping them separate, we find focused communities. Looking at the big communities found by the GA algorithm, for the Facebook and Twitter dataset, we observe that one can easily identify the focused communities related to *sports*, *politics*, *news*, and so on.

In this section, we therefore present a new approach that can further divide these big communities into small focused communities. Note that while extracting small communities from each of the big communities, we only 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. At this moment 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 this approach the incremental community extraction algorithm, denoted by Incre-Comm-Extraction(INC).

Our algorithm works in a recursive fashion. At the beginning of every round, we call the GA algorithm and for each community that the GA algorithm outputs, our algorithm either declares that as a final community or recalls our algorithm recursively for that community to divide it further. 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 reaches size s, an upper bound on the community size, which is an input parameter. The details of INC algorithm are outlined in Algorithm 1.

Algorithm 1 Template for Incremental Community Extraction (INC) algorithm.

```
1: procedure Incre-Comm-Extraction(G_r)
2:     C' ← GreedyAgglomerative(G_r)
3:     if |C'| = 1 then
4:         Let c_1 be the only community.                ▷ c_1 = V(G_r)
5:         C ← C ∪ c_1
6:         return
7:     c' ← ∅
8:     for each community c_i ∈ C' do
9:         if |c_i| = 1 then
10:            c' ← c' ∪ c_i
11:        else if |c_i| ≤ s then                        ▷ Optional feature
12:            C ← C ∪ c_i
13:        else
14:            G_i ← G(V(c_i), E(c_i))
15:            Incre-Comm-Extraction(G_i)
16:    if |c'| ≠ 0 then
17:        C ← C ∪ c'
```

Note that the idea of INC algorithm is similar to [18] (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 [18]. 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. Although our algorithm is hierarchical in nature, this is not to be confused with [6], which uses similarity metric as distance metric and then applies agglomerative hierarchical clustering algorithm to find clusters or communities.

Based on the complexity of CNM and INC algorithm (details are skipped), the INC algorithm may call GA algorithm n times in worst case. But, our experiments show that the ratio of the time taken by INC algorithm and GA algorithm is much smaller than n. 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.

4. RESULTS AND DISCUSSION

Our experiments were performed on an Intel(R) Xeon(R) E7540 processor running at 2 GHz. Our algorithms were implemented in C and compiled with GCC version 4.4.5 using the -O3 flag.

Table 3 shows the number of communities found by INC and 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 decreases. Most of the cliques have size less than 5. CNM algorithm was not able to identify these small size cliques as can be seen the frequencies of small sized communities is not high, rather it generates large communities. This is because CNM algorithm merges most of cliques to form a large community (sizes 73, 103, and 128 for *tw_dyn* and sizes 71 and 184 for *tw_stat*) to maximize the 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 is higher than large size communities. We observed similar results for frequencies of community and clique size in Facebook network.

Table 3: 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	0	50	27	0	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	-	-	-	-	-	-	-	-	-	-

Next, we showcase the extracted communities using our algorithm. Usually the communities found by community extraction algorithms are represented as dendrograms [2, 5, 10]. Since the dendrograms are quite big in our case, we present partial dendrograms both for Facebook (Figure 2) and Twitter (Figure 1). Each internal node in the dendrograms represents a community whereas each leaf node is a *wall/profile*. The full dendrograms of all the above mentioned graphs can be found at our website[3]. 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.

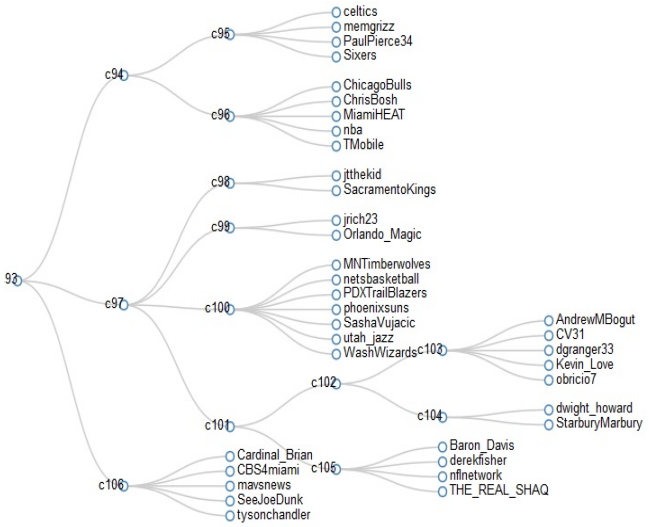

Figure 1: NBA Basketball - Dynamic.

In Figure 1, partial dendrogram displays communities from the Twitter dataset for the category *NBA Basketball*. Our User-Interest based approach is able to closely model the explicit static network (not shown) and finds more affiliations based on the user interests i.e. the dynamic network is able to capture more affiliations that may not similarly labeled but if looked closely they were indirectly affiliated. For example, in Figure 1, TMobile is captured into community c_{96} because TMobile hosted many events at the 2011 NBA All-Star Game[4]. This is a very interesting result because dynamic networks are constantly changing, we can capture affiliations which are temporal in nature and thus strengthening the case for viewing dynamic networks compared to static networks. Therefore, this study can help businesses to identify target communities for marketing purposes.

In Figure 2, we look at a partial branch of the dendrogram of Facebook network represented by node c_{230}. Our approach has successfully extracted several focused communities that belong to categories such as *Technology*(c_{231}), *Consumer Merchandize*(c_{232}), *Retail*(c_{243}), *Travel & Leisure*(c_{244}), *Food* (c_{248}) and *Baby Products*(c_{251}). We got many more interesting focused communities, which can be found at[3]. Note that most of these focused communities found by our INC algorithm belong to one large community using CNM algorithm, which always tries to maximize the modularity.

As mentioned previously, modularity might not reflect the right community structure, we therefore use *modularity density*, introduced in [17], as a metric to compare the quality between CNM and INC algorithm. Modularity density is defined as the sum over the clusters of the ratio between the difference of the internal and external degrees of the

[3]pulse.eecs.northwestern.edu/∼drp925/inc/graph.php

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

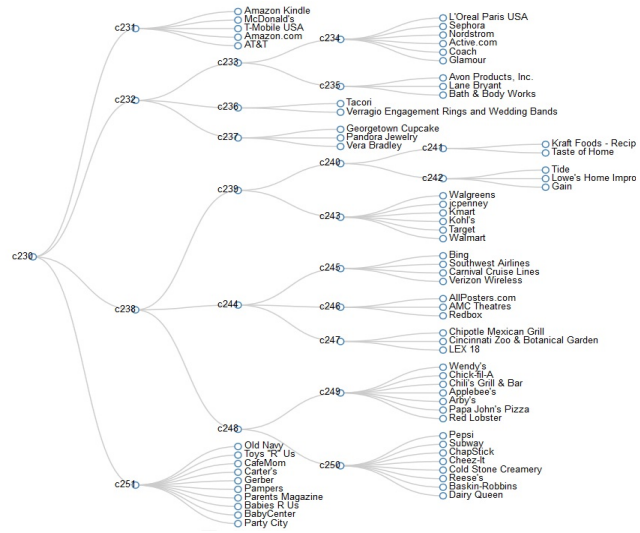

Figure 2: Partial dendrogram showing communities in Facebook

Table 4: Comparison between CNM and INC

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

cluster and the cluster size. Table 4 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 CNM algorithm compared to INC algorithm, whereas mod_den is higher (or similar) for INC algorithm than CNM algorithm. As modularity density delivers better results than modularity [3], INC algorithm recovers natural communities from the social networks compared to CNM algorithm which extracts large communities to maximize modularity.

5. CONCLUSION

In this paper, we present a new way of modelling social networks. 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 find 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 target communities in the context of business requirements. In the future we intend to experiment with time based user-interest modelling, to study effects on the community structure with temporal events and develop a clique based community extraction algorithm that allows single user to belong to multiple communities.

6. ACKNOWLEDGMENT

This work is supported in part by NSF award numbers CCF-0621443, OCI-0724599, CCF-0833131, CNS-0830927, IIS-0905205, OCI-0956311, CCF-0938000, CCF-1043085, CCF-1029166, and OCI-1144061, and in part by DOE grants DE-FG02-08ER25848, DE-SC0001283, DE-SC0005309, DE-SC0005340, and DE-SC0007456.

7. REFERENCES

[1] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley. Classes of small-world networks. *PNAS*, 97(21):11149–11152, 2000.

[2] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111, 2004.

[3] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[4] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36, 2007.

[5] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[6] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[7] J. Leskovec. Stanford network analysis project. http://snap.stanford.edu/, 2009.

[8] L. Leydesdorff. On the normalization and visualization of author co-citation data: Salton's cosine versus the jaccard index. *Journal of the Amer. Society for Info. Science and Technology*, 59(1):77–85, 2008.

[9] K. Madduri. Snap: Small-world network analysis and partitioning. http://snap-graph.sourceforge.net, 2008.

[10] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69(6):066133, 2004.

[11] M. E. J. Newman. Modularity and community structure in networks. *PNAS*, 103(23), 2006.

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

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

[14] S. H. Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2001.

[15] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, Cambridge, 1994.

[16] F. Wu and B. A. Huberman. Finding communities in linear time: A physics approach. *The European Physical Journal B*, 38:331–338, 2004.

[17] S. Zhang, X. Ning, and C. Ding. Maximizing modularity density for exploring modular organization of protein interaction networks. *Symp. on Opt. and Systems Biology*, pages 361–370, 2009.

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