Detecting Communities in Social Networks using Max-Min Modularity

Jiyang Chen* Osmar R. Zaïane* Randy Goebel *

Abstract

Many datasets can be described in the form of graphs or networks where nodes in the graph represent entities and edges represent relationships between pairs of entities. A common property of these networks is their community structure, considered as clusters of densely connected groups of vertices, with only sparser connections between groups. The identification of such communities relies on some notion of clustering or density measure, which defines the communities that can be found. However, previous community detection methods usually apply the same structural measure on all kinds of networks, despite their distinct dissimilar features. In this paper, we present a new community mining measure, Max-Min Modularity, which considers both connected pairs and criteria defined by domain experts in finding communities, and then specify a hierarchical clustering algorithm to detect communities in networks. When applied to real world networks for which the community structures are already known, our method shows improvement over previous algorithms. In addition, when applied to randomly generated networks for which we only have approximate information about communities, it gives promising results which shows the algorithm's robustness against noise.

1 Introduction

Many datasets can be described in the form of graphs or networks where nodes in the graph represent entities and edges represent relationships between pairs of entities. For example, the World Wide Web (WWW) can be viewed as a very large graph where nodes represent web pages and edges represent hyperlinks between pages. In social networks, nodes typically represent individuals and edges indicate relationships, e.g., in a tele-communication network, each node is a phone number and edges represent the fact that two nodes communicated. A common property of these networks is their *community structure* which notes the existence of densely connected groups of vertices, with only sparser connections between groups. *Community Mining*, which focuses on the detection and characterization of such network structure, has received considerable attention over the past few years in sociology and lately data mining. A community can be defined as a group of entities that share similar properties or connect to each other via selected relations [45]. Identifying these connections and locating entities in different communities is the main goal of community mining research.

The ability to detect communities could be of significant practical importance. For example, groups of web pages that link to more web pages in the community than to pages outside might correspond to sets of web pages on related topics, which can enable search engines and portals to increase the precision and recall of search results by focusing on narrow but topically-related subsets of the web [12]; groups within social networks might correspond to social communities, which can be used to understand the data, such as organization structures, academic collaborations and the communities in tele-communication networks.

Traditional data mining algorithms, such as association rule mining, supervised classification and clustering analysis, commonly attempt to find patterns in a data set characterized by a collection of independent instances, which is consistent with the classical statistical inference problem of trying to identify a model given an independent, identically distributed (IID) sample [14]. However, a new emerging challenge that data mining researchers face is solving the community mining problem on richly structured, heterogeneous data sets. Such data sets are usually modeled as networks or graphs and contain multiple object types, which can be related to each other in various ways, e.g., commercial data describing relations between customers, products and transactions. Naively applying traditional statistical inference procedures, which assume that instances are independent, can lead to inappropriate conclusions about the data [21], for example, for a search engine, indexing and clustering web pages based on the text content without considering their linking structure would definitely lead to unsatisfactory results for queries. Various relation-based methods have been developed, such as the spectral clustering approaches [10, 36, 41] and modularity-based algorithms [7, 30]. However, none of them distinguish the intrinsic features of the domain of the network in question. In other words, the same

*Department of Computing Science, University of Alberta, {jiyang, zaiane, goebel}@cs.ualberta.ca

structural measure has been applied to all kinds of networks, despite their different characteristics, which can be achieved from domain expertise.

In this paper, we present a new modularity-based measure we call Max-Min Modularity, which considers the property of both connected and user-defined related node pairs in finding communities. We use it to estimate the compatibility between the discovered communities and the link structure. Generally speaking, our Max-Min Modularity compares the difference between the fraction of links that occur within communities to the fraction that would be expected to occur if the links were randomly distributed; in addition, it also considers the fraction of user-defined related node pairs within communities to the expected fraction of such pairs. Based on this measure, we propose a hierarchical clustering algorithm to detect communities in networks. Our work makes the following contributions:

- We propose a method to include domain knowledge as guiding criteria in the community detection process by either rewarding or penalizing the metric that evaluates the discovered structure, without increasing the algorithm complexity.
- Our new measure and algorithm improves the accuracy for community detection over previous algorithms when applied to real world networks for which the community structures are already known, and also gives promising results when applied to randomly generated networks for which we only have approximate information about communities. This shows the robustness of the algorithm against noise.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 introduces definitions of the modularity evaluation. We propose our Max-Min modularity measure and community mining algorithm in Section 4 and report experimental results in Section 5. Possible extensions are discussed in Section 6, followed by conclusions in Section 7.

2 Related Work

Generally speaking, we can divide previous research of finding groups in networks into two main principle lines of research: *graph partitioning* and *hierarchical clustering*. These two lines of research are really addressing the same question, albeit by somewhat different means. There are, however, important differences between the goals of the two camps that make quite different technical approaches desirable [29]. For example, *graph partitioning* approaches usually know in advance the number and size of the groups into which the network is to be split, while *hierarchical clustering* methods normally assume that the network of interests divide naturally into some subgroups, determined by the network itself and not by the user.

Graph Partitioning. Generally, finding an exact solution to a partitioning task is believed to be an NP-complete problem, making it prohibitively difficult to solve for large graphs. However, a wide variety of heuristic algorithms have been developed and give good solutions in many cases, e.g., multilevel partitioning [22], k-partite graph partitioning [24], relational clustering [25], flow-based methods [12], information-theoretic methods [9] and spectral clustering [33]. The main problem for these methods is that input parameters such as the number of the partitions and their sizes are usually required, but we do not typically know how many communities there are, and there is no reason that they should be roughly the same size. Various benefit functions have been proposed to avoid the problem, such as the *normalized cut* [36] and the *min-max cut* [10]. However, these approaches are biased in favor of divisions into equal-sized parts and thus still suffer from the same drawbacks that make graph partitioning inappropriate for community mining.

In the field of theoretical computer science, correlation clustering [2, 5, 37] considers a complete graph on n vertices, where each edge (u, v) is labeled either $+$ or $-$ depending on whether u and v have been deemed to be similar or different. Similar to our problem, the goal is to find a partition that agrees as much as possible with the edge labels, i.e., a clustering that maximizes the number of $+$ edges within clusters and minimize the number of $-$ edges inside clusters. However, while correlation clustering assumes the graph is complete and each connection is either positive or negative, such assumption is not true for community detection, where graphs are usually sparse and many of the edges are unobserved, i.e. labeled as 0. Moreover, while existing methods [15, 16] for correlation clustering require the user to specify parameters that are usually hard to determine [1], e.g., the number of clusters, our algorithm does not require parameters.

The idea of considering domain knowledge as related/unrelated pairs in this paper is analogous to the notions of must/cannot links in semi-supervised clustering [3, 39, 40]. However, in semi-supervised clustering, the labeled data is used for cluster initialization [3] and the link constraints must be satisfied [39, 40]. Moreover, the number of clusters k is usually required as the starting parameter. On the other hand, our algorithm does not require parameters and the domain knowledge in our work is used to guide the bottom-up hierarchical clustering process, instead of generating ini-

tial communities. The given related/unrelated pairs are not enforced to be in the same/different communities, but contribute in calculating a metric score which evaluates the "closeness" of two communities.

Hierarchical Clustering. The main idea of this technique is to discover natural divisions of social networks into groups, based on various metrics of similarity (usually represented as similarity x_{ij} between pairs (i, j) of vertices). The hierarchical clustering method has the advantage that it does not require the size or number of groups we want to find beforehand, therefore, it has been applied to various social networks with natural or predefined similarity metrics, such as the modularity and betweenness measure [7, 17, 27, 30]. However, they are usually slow and the performance highly depends on the corresponding metrics.

Recently, real world networks have been shown to change over time and have an overlapping community structure, which is hard to grasp with classical clustering methods where every vertex of the graph belongs to only one community. Based on these observations, fuzzy methods [18, 26, 32, 46] and dynamic approaches [4, 38] have been proposed for overlapping structure and dynamic community detection. Recent work by Xu et al. [42] proposed a fast SCAN algorithm to detect not only clusters, but also hubs and outliers in networks. However, the performance of their approach highly depends on input parameters, which are very sensitive. While all these methods successfully find communities, none of them are able to use domain knowledge as clustering criteria to help the mining process.

3 Preliminaries

As reviewed above, many algorithms are able to detect communities. However, ground truth is difficult to come by, thus validation becomes an issue when using these algorithms on unknown networks. How do we know whether the communities discovered by the algorithms are truly good ones? Since community mining algorithms can produce communities even for completely random networks which have no meaningful community structure, how can we measure the structure that is found for these "structureless" networks is an important question to answer. To solve this problem, computer scientists proposed several benefit functions based on cut sizes, e.g., *normalized cut* [36]. However, cut sizes do not accurately reflect the intuitive concept of social network communities and thus are the wrong measure to optimize. A good division of a network into communities is not merely one in which the number of edges running between groups is small. Rather, it is one in which the number of edges between groups is *smaller than expected* [28], which is the intuition behind the modularity measure.

3.1 The Modularity Measure The modularity Q is proposed by Newman and Girvan [30] as a measure of the quality of a particular division of a network, and is defined as follows:

$$Q = (\textit{number of edges within communities}) - (\textit{expected number of such edges}) \text{ [30]}$$

The basic idea is to compare the division to a "null model", a randomized network with exactly the same vertices and same degree, in which edges are placed randomly without regard to community structure. More precisely, consider a particular division of a network into k communities, the division can be represented by a $k \times k$ symmetric matrix e, of which each element e_{ij} is the fraction of all edges in the network that link vertices in community i to vertices in community j. The matrix trace $Tr(e) = \sum_i e_{ii}$ gives the fraction of edges in the network that connect vertices in the same community, and clearly a good community division should have a high trace value. However, the trace alone is not a good indicator of the quality, since placing all vertices in one single community would give the maximal value 1 while providing no information of the community structure. The row sum $a_i = \sum_j e_{ij}$ represents the fraction of edges that has at least one end in community i. So, a_i^2 is the expected fraction of edges within community i if the edges were distributed randomly on the null model network. Thus, the modularity measure is defined by:

$$Q = \sum (e_{ii} - a_i^2) = Tr(e) - \|e^2\| \tag{3.1}$$

where $\|x\|$ indicates the sum of the elements of the matrix x. In other words, the modularity Q measures the fraction of the edges in the network that connect vertices of the same type, i.e., within-community edges, minus the expected value of the same quantity in a network with the same community division but with random connections between the vertices [17]. If the number of within community edges is no better than random, $Q = 0$. Values of Q that are close to 1, which is the maximum, indicates strong community structure. Q typically falls in the range from 0.3 to 0.7 [17] and high values are rare.

3.2 Drawbacks of Modularity Q The modularity approach has been pursued by a number of authors [11, 19, 28, 41], and has been proved highly effective in practice for community evaluation [8]. However, there are three major problems for the Q measure. At first, the modularity requires information of the entire structure of the graph, which is problematic for huge networks like the WWW. To solve this problem, Clauset [6]

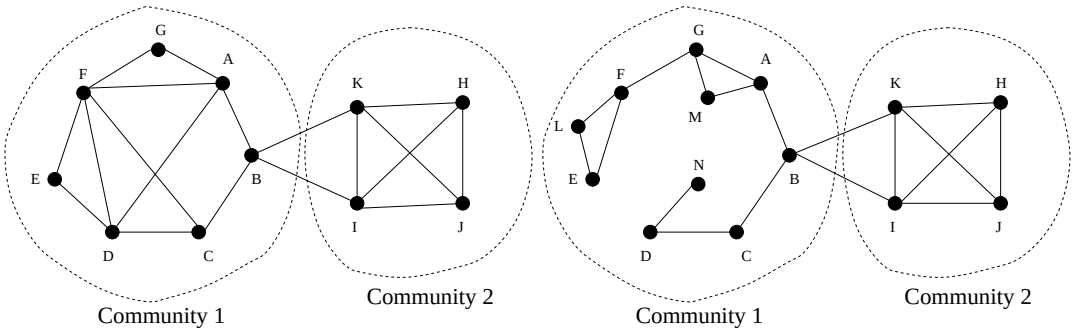

Figure 1: Two Network Examples with same Modularity score, but the right network has more absent links (disconnected node pairs in community) than the left network.

proposed a measure of local community structure, called *local modularity*, for graphs which lack global knowledge. Secondly, recent research showed that modularity-based methods have resolution limit and may fail to identify communities smaller than a certain scale [13]. Possible solutions include recursive algorithms based on modularity optimization [34]. At last, as pointed out by Scripps et al. [35], the modularity only measures existing links on the network, but does not explicitly consider the absent links between two nodes in the same community. In other words, the modularity only measures how good the discovered community structure fits the existing links (connected vertices should be in the same community), but fails to measure how good the structure fits the absent links (disconnected vertices might not be necessarily in the same community). For example, the two networks shown in Figure 1 have the same number of edges and the same Q, which is 0.360, but the intuition is that the community division in the second network is worse, since it has more disconnected node pairs within community 1, which is not considered by the Q measure. Therefore, modularity fails to compare the community structure between different graphs. To solve this problem, Scripps [35] proposed two ratios p and q, measuring the fraction of links within communities and absent links between communities, respectively. The drawback of their method is that the interpretation is not clear since there are two measures: a mining result can have higher p but lower q than the other, which makes it hard to compare the quality of different community structures. In this paper, we propose a new evaluation measure to increase the accuracy of community detection, which not only solves this particular problem by taking unrelated node pairs (defined by domain knowledge) into consideration, but also makes it possible to compare the community structure quality between different graphs.

4 Our Proposed Elaboration

Communities are defined to be densely connected groups of entities in a relational network, i.e., nodes in a strong community should be related to all other nodes in the same community. However, the original modularity measure does not consider the absent links. In other words, it only checks whether connected vertices are placed in the same community, but ignores disconnected vertices that share the same community. Therefore, to more thoroughly evaluate a network division, we should not only reward the evaluation score if connected vertices are put in the same community, but also penalize the score if disconnected vertices are in the same community. However, a "disconnection" could possibly be an unobserved connection, which is very common in biological and social networks, so it is dangerous to assume disconnection to be a negative sign of the community structure. Therefore, while connected pairs remain as positive signs of a strong community structure, we separate the disconnected pair set into two categories based on knowledge provided by domain experts: the *related pair set*, in which pairs of nodes are *possibly* related, and *unrelated pair set*, in which pairs of nodes are *certainly* unrelated. We only penalize our measure score if we see unrelated pairs share the same community. Based on this criterion, we propose a user-defined community structure measure, we call *Max-Min (MM) Modularity*.

The idea of MM Modularity is based on the intuition that a good division of a network into communities is not merely one in which the number of edges between groups is smaller than expected, it is also one in which *the number of unrelated pairs within groups is smaller than expected*. Only if both the numbers of between-group edges and within-group *unrelated pairs* are significantly lower than would be expected purely by chance, can we justifiably claim to have found significant community structure. Equivalently, we can exam-

ine the number of edges within communities and *unrelated pairs* between communities and look for divisions of the network in which this number is higher than expected. These two approaches are equivalent since the total number of edges/pairs is fixed and any edges/pairs that do not lie between communities must necessarily lie inside one of them [28].

Generally speaking, our evaluation attempts to maximize the number of edges within groups and minimize the number of *unrelated pairs* from the user-defined unrelated pair set within groups at the same time, therefore we named it *Max-Min Modularity*. Note that maximizing the edge number within groups does not automatically minimize the unrelated pair number, e.g., if we have no network knowledge, thus have no *related pairs*, and *unrelated pairs* as disconnected node pairs, consider a node that only connects one member of a community with size n, maximizing the within-group edge number by including that node in this community would increase the unrelated pair number by $n-1$.

4.1 Generalizing the Max-Min Modularity The modularity Q can be transformed from its original form, which is community-based, to a node-based form. Given a unweighted and undirected network $G = (V, E)$, $|V| = n, |E| = m$, let A_{xy} be an element of the adjacency matrix of G:

$$A_{xy} = \begin{cases} 1 & \textit{if vertices } x \textit{ and } y \textit{ are connected} \\ 0 & \textit{otherwise} \end{cases}$$

Assume the network is divided into k communities and node x belongs to community C_x, the fraction of edges that fall between community i and community j is defined as follows:

$$e_{ij} = \frac{1}{2m} \sum_{xy} A_{xy} \phi(C_x, i) \phi(C_y, j)$$

where the ϕ function $\phi(i, j)$ is 1 if i and j are the same community and 0 otherwise. The degree d_x of a vertex x is the number of edges that connect to it: $d_x = \sum_y A_{xy}$. Therefore, the fraction of edges that have at least one end in community i is:

$$\begin{aligned} a_i &= \tfrac{1}{2m} \textstyle\sum_{xy} A_{xy} \phi(C_x, i) \\ &= \tfrac{1}{2m} \textstyle\sum_x d_x \phi(C_x, i) \end{aligned}$$

From all the equations above:

$$\begin{aligned} Q &= \textstyle\sum_i (e_{ii} - a_i^2) \\ &= \textstyle\sum_i [\tfrac{1}{2m} \sum_{xy} A_{xy} \phi(C_x, i) \phi(C_y, i) \\ &\quad - \tfrac{1}{2m} \textstyle\sum_x d_x \phi(C_x, i) \tfrac{1}{2m} \sum_y d_y \phi(C_y, i)] \\ &= \tfrac{1}{2m} \textstyle\sum_{xy} [A_{xy} - \tfrac{d_x d_y}{2m}] \sum_i \phi(C_x, i) \phi(C_y, i) \end{aligned}$$

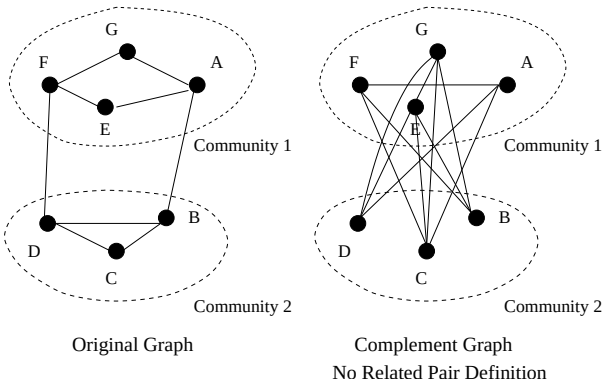

Figure 2: A Graph Division and its Complement

Define $P_{xy} = \frac{d_x d_y}{2m}$, we have the modularity Q as follows:

$$Q = \frac{1}{2m} \sum_{xy} [A_{xy} - P_{xy}] \phi(C_x, C_y) \tag{4.2}$$

We see that the original modularity has already measured the first part of MM Modularity:

$$Q_{max} = \frac{1}{2m} \sum_{xy} [A_{xy} - P_{xy}] \phi(C_x, C_y) \tag{4.3}$$

thus we only need to measure the other part, which is minimizing the *unrelated pair* fraction within communities. It is obvious that, if a division contains very few unrelated node pairs within communities for a graph, the same division will have equivalently few connected node pairs within communities for the complement graph, which is a graph on the same vertices as the original network such that two nodes are connected if and only if they are defined as an *unrelated pair* in the original graph. In other words, the better this division is for the original network regarding *containing few unrelated node pairs within communities*, the worse it is for the complement graph as a community structure since *there are equally few connected node pairs within communities* (See Figure 2). Therefore, we can compute the modularity score for a division on the complement graph of the network. The lower Q score we get, the better community division we have for the original network. Note that, although building a complement graph can be very expensive especially when the original graph is sparse, our method does not require extra computation or materialization of this complement graph, since we only need to maintain and update the *related pair* set defined by domain experts. Other information we need, which is the node degree and number of edges, can be easily achieved from the structure of the original graph.

More precisely, given an unweighted graph $G = (V, E)$, $V = \{v_x | 1 \le x \le n\}$, $E = \{e_x | 1 \le x \le m\}$

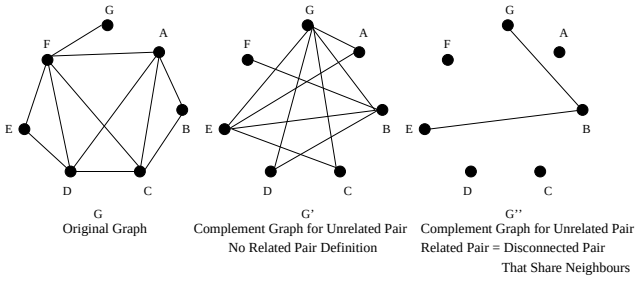

Figure 3: Building Complement Graphs

and the user-defined criteria U to define whether two disconnected nodes i, j are *related* $(i,j) \in U$ or *unrelated* $(i,j) \notin U$, we create $G' = (V, E')$, such that if and only if $(i,j) \notin E$ and $(i,j) \notin U, (i,j) \in E'$, i.e., G' is G's complement graph. A' is the adjacency matrix of G'': $A'_{ij} = 1$ iff $A_{ij} = 0$ and $(i,j) \notin U$. We define Q_{min} as

$$Q_{min} = \frac{1}{n(n-1) - 2m - 2|U|} \sum_{xy} [A'_{xy} - P'_{xy}] \phi(C_x, C_y) \quad (4.4)$$

$\phi(C_x, C_y) = 1$ if C_x and C_y are the same community, 0 otherwise. $|U|$ is the number of pairs in U. Similarly, P'_{xy} is the expected probability of an edge between vertices x and y in a random graph:

$$P'_{xy} = \frac{(d'_x)(d'_y)}{2m'}$$

where d'_x is the degree of node x in G', and $m' = \frac{n(n-1) - 2m - 2|U|}{2}$. Since $d'_x = n - d_x - u_x$, we have

$$P'_{xy} = \frac{(n - d_x - u_x)(n - d_y - u_y)}{n(n-1) - 2m - 2|U|}$$

where u_x and u_y are number of nodes that are disconnected from but defined as *related* nodes with x and y.

Now we want to maximize Q_{max} and minimize Q_{min} at the same time. Fortunately, it can be achieved by the following equation:

$$\begin{aligned} Q_{Max\_Min} &= Q_{max} - Q_{min} \\ &= \textstyle\sum_{xy} [\frac{1}{2m}(A_{xy} - P_{xy}) \\ &\quad -\frac{1}{2m'}(A'_{xy} - P'_{xy})] \phi(C_x, C_y) \end{aligned}$$

The higher Q_{Max_Min} is, the better community division we get. Note that we choose to use $Q_{max} - Q_{min}$ instead of $\frac{Q_{max}}{Q_{min}}$ for Q_{Max_Min} since the former equation allows us to compute the Δ modularity between every pair of nodes. Also note that it is easy to extend the MM Modularity for weighted graphs by using the weight in degree computation for Q_{max} and Q_{min}.

We present an example for user-defined criteria in the following. In social networks, the neighbourhood around nodes are usually as important as direct connections. Thus we naturally define disconnect people as *related pair* if they connect to the same intermediary person. Therefore, we define U as if $(i,j) \notin E$ and there is such k that $(i,k), (k,j) \in E$, we have $(i,j) \in U$. By applying this criterion, we reward the MM modularity for connected pairs in the same community, penalize it for pairs that are in same communities and has no shared neighbour, and do not reward or penalize pairs that are disconnected but share neighbour nodes. An example for building complement graphs for this criterion is shown in Figure 3. Other constraints can also be applied to define *related pairs*, as discussed in Section 6.

ALGORITHM 4.1. Hierarchical Clustering Algorithm HMaxMin to greedily optimize Q_{Max_Min}

Input: A social network $G = (V, E)$, $V = \{v_i | 1 \leq i \leq n\}$, $E = \{e_x | 1 \leq x \leq m\}$, Adjacency Matrix A, the user-defined criteria U.
Output: A division of V: $C_1, C_2, ...C_k$.
1. Assume each node is the sole member of one of the n communities. Build related pair matrix S, S_{ij} equals the number of related pairs between community i and j:
$S_{ij} = 1$ iff $(i,j) \in U$.
2. Compute the symmetric sparse matrix containing ΔQ_{xy} for each connected pair x, y:
$\Delta Q_{xy} = \frac{1}{2m}(1 - P_{xy}) - \frac{1}{2m'}(0 - P'_{xy})$.
Save each matrix row both as a balanced binary tree t_x and as a max-heap h_x. Save the largest element of each matrix row along with the x y label in a max-heap H. For each node x, we set: $a_x = \frac{d_x}{2m}$, $a'_x = \frac{d'_x}{2m'}$.
3. While ($pop\_heap(H) > 0$)
Select the largest element H and the corresponding x, y. Update ΔQ by merging y row (column) into x row (column) such that:
if community z connect to both x and y in G:
$\Delta Q_{xz} = \Delta Q_{xz} + \Delta Q_{yz}$
if z only connect to x in G and connect to y in G':
($|y|$ equals the number of nodes in community y)
$\Delta Q_{xz} = \Delta Q_{xz} + 2a'_y a'_z - 2a_y a_z - \frac{2|y||z|}{m'} + \frac{2S_{yz}}{m'}$
if z only connect to y in G and connect to x in G':
$\Delta Q_{xz} = 2a'_x a'_z - 2a_x a_z - \frac{2|x||z|}{m'} + \frac{2S_{xz}}{m'} + \Delta Q_{yz}$
Mark y row (column) in ΔQ and S as merged.
Update S by adding y row (column) into x column.
Update $new\_a_x = a_x + a_y, new\_a'_x = a'_x + a'_y$
Update t_x, update all h_z, update H.
4. Label merged nodes in the same unmarked row to be in the same community, as $C_1, C_2, ...C_k$.
5. Return $C_1, C_2, ...C_k$.

4.2 Algorithm for Community Detection Here we propose a new method to evaluate the quality of the discovered community structure based on the MM Modularity. We may consider, if a high value of Q_{Max_Min} represents a good community division, one can simply optimize Q_{Max_Min} over all possible divisions to find the best one. However, to find the optimal value of Q_{Max_Min} is very costly: to carry out a complete search of all possible divisions for the optimal value of Q_{Max_Min} would take at least an exponential amount of time, and is thus infeasible for large networks. Therefore, we propose a hierarchical clustering algorithm HMaxMin (see Algorithm 4.1) to greedily optimize the MM Modularity to find the approximate optimal value.

Recall that we reward the modularity for connected pairs, penalize it for unrelated pairs, and do nothing for disconnected but related pairs. Merging a pair of nodes between which there are no edges at all can never result in an increase in Q_{Max_Min}, thus in step 2 we only compute and store modularity scores for connected pairs in a sparse matrix. In step 3, we greedily merge the pair of communities which provides the highest modularity gain into one community and update the modularity matrix as well as the related pair matrix. For updating the modularity matrix, we first treat all pairs between community i and j as unrelated, i.e., $|i| * |j|$ pairs, then add the extra deducted value for related pairs (note that $Q_{i_r j_r} = \frac{1}{2m}(0 - P_{ij}) - \frac{1}{2m'}(0 - P'_{ij})$ if i and j are related and $Q_{ij} = \frac{1}{2m}(0 - P_{ij}) - \frac{1}{2m'}(1 - P'_{ij})$ if they are unrelated, thus $Q_{i_r j_r} = Q_{ij} + \frac{1}{2m'}$).

For algorithm complexity, consider we have n nodes and m edges. In step 2 we only need to consider those pairs connected by edges, of which there will be at any time at most m. If we use n_x to represent the number of neighboring communities of community x, we have n_x elements for the x row in the sparse matrix. In step 3 of Algorithm 4.1, since we need to merge y row into x row, we will have $n_x + n_y$ insertions in the worse case. Since the rows are stored as balanced binary trees, each of the insertions take $O(\log n)$ in the worst case. Therefore, updating the matrix in step 2 takes $O((n_x + n_y) \log n)$ time. Similarly, we only need to update the heap for the k row if community k is adjacent to community x or y. We need to do at most $n_x + n_y$ updates of h_k, each of which takes $O(\log n)$ time, for a total of $O((n_i + n_j) \log n)$ time. Updating the related pair matrix S takes $O(|S_y|)$ (S_y is the number of elements in the y row of S), however, we can always choose y so that $|S_y| \leq |S_x|$, therefore, updating S takes $O(1)$ time. Since each merge takes $O((n_i + n_j) \log n)$ time, the total running time is the $O(\log n)$ times the sum over all degrees of the merged communities along the dendrogram. In the worst case, each node would contribute its degree to all of the communities it belongs to, along the path in the dendrogram from the node to the root, which makes the total degree $2m$, therefore, if the dendrogram has depth D, the algorithm runs in $O(mD \log n)$ time in a sparse graph. The approximate optimal Q value is also accumulated as the algorithm goes along. Thus, our algorithm includes domain knowledge but still runs in the same complexity as the similar algorithm proposed in [7].

5 Experiment Result

In this section, we apply our MM Modularity and the HMaxMin algorithm to detect communities on various social networks, including several data sets collected from real networks with ground truth as well as a synthetic data set which is randomly generated with given parameters, such as the graph size and the number of communities. The domain knowledge used is the example presented in Section 4.1: *if $(i, j) \notin E$ and there is such k that $(i, k), (k, j) \in E$, we have $(i, j) \in U$.* All the experiments were conducted on a PC with a 3.0 GHz Xeon processor and 4GB of RAM.

5.1 Scalability To test our algorithm on large datasets, we ran our algorithm on the largest component of the collaboration network of scientists posting preprints at www.arxiv.org [27], which has 27,519 nodes and 116,181 edges, and the IMDB network between actors who share involving movies [20], which has 47,436 nodes and 379,196 edges, within 376 and 1,037 seconds, respectively. To further evaluate the algorithm efficiency, we generated ten random graphs of vertices ranging from 10,000 to 500,000 and the number of edges ranging from 20,000 to 1,000,000. Figure 4 shows the performance of our algorithm on those networks. It clearly reflects the $O(mD \log n)$ complexity of our approach.

However, we do not have ground truth to validate our result for such large datasets, thus we turn to synthetic data and real world datasets to evaluate the accuracy of our algorithm. Danon et al. [8] found that the modularity method outperformed all other methods for community detection of which they were aware, in most cases by an impressive margin. Therefore, we compared our HMaxMin algorithm with a similar hierarchical clustering algorithm [7] (we refer to it as algorithm N), which uses Newman's modularity to measure community structure, to show that our MM Modularity is more accurate for community detection tasks by including domain knowledge.

5.2 Evaluation Approach To evaluate how closely each community in the result matches its corresponding

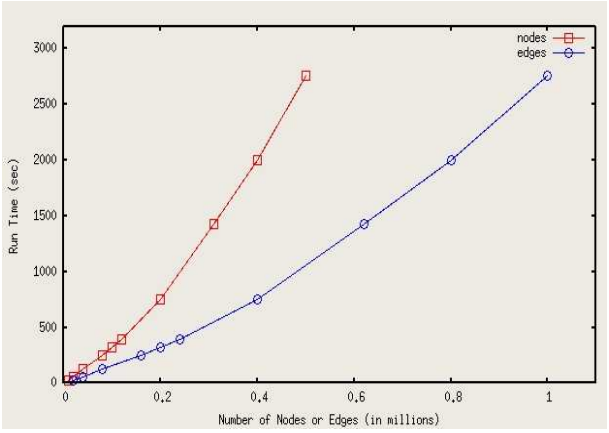

Figure 4: Algorithm Running Time

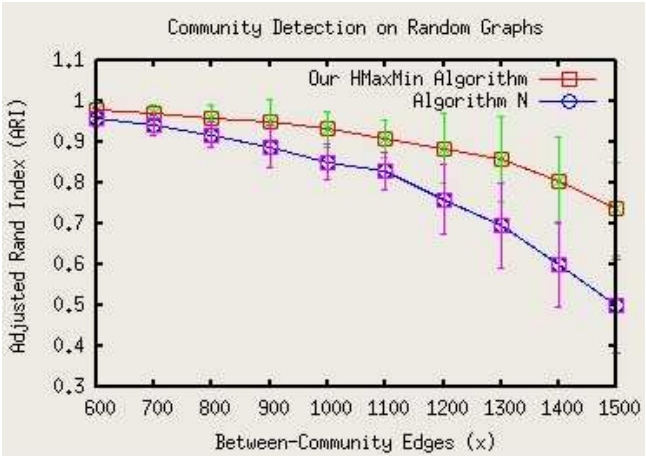

Figure 5: Synthetic Data Results (each point is an average over 50 1,000-node graphs.)

community in ground truth, we adapt the Adjusted Rand Index (ARI) [43] as the performance metric for accuracy. The ARI measures how similar are the partition of objects according to the real communities (R) and the partition in an algorithm result (P). Denote a, b, c and d as the number of object pairs that are in the same community in both R and P, in the same community in R but not in P, in the same community in P but not in R, and in different communities in both R and P, respectively. ARI is defined as follows.

$$ARI(R,P) = \frac{2(a*d-b*c)}{(a+b)*(b+d)+(a+c)*(c+d)}$$

The more similar the two partitions (larger a and d, smaller b and c), the larger the ARI value. ARI will be 1 if R and P are identical and 0 if P is a random partition for the graph.

5.3 Synthetic Data To test the performance of our algorithm on networks with varying degrees of community structure, we have applied it to a large set of randomly generated graphs. Each graph was constructed with 1000 vertices and 5 communities, each of which had 200 vertices. At first, each vertice was connected to 6 other randomly chosen nodes in the same community and had no connection to nodes in the different communities, thus we get 3000 within-community edges. Then we added a number of between-community edges, x, into the graph. Both ends were randomly chosen and each node could only connect up to 4 nodes in different communities so that within-community connections for all nodes were supposed to be stronger than between-community connections. This produces graphs which have a known community structure, but are essentially random in other respects. Moreover, we can control the "noise", i.e., between-community edges, by adjusting x. Similar synthetic data generation methods have been used in [6, 17].

Using these graphs, we tested and compared the performance of HMaxMin algorithm and algorithm N with different x, as shown in Figure 5. As the figure shows, HMaxMin performs well, with a > 0.7 average ARI on graphs with less than 1500 between-community edges (50% of the within-community edges). HMaxMin begins to "fail" when x exceeds 1500, however, communities may not exist in such circumstances. On the same plot, we also show the performance of the algorithm based on the original modularity Q (Algorithm N) and, as we can see, the algorithm performs measurably worse than our algorithm. Interestingly, the performance of the two algorithms are about the same when the community structure is clear and strong, but when we increase the noise edge numbers, the accuracy of HMaxMin drops much slower than algorithm N. Therefore, it is reasonable to believe that our algorithm is more robust in finding community structure for data with considerable noise.

5.4 The Karate Club While random mid-size networks provide a reproducible and well-controlled testbed for community discovery evaluation, it is also desirable to test and compare the performance of our algorithm on real world networks. Since ground truth of large datasets is hard to come by, we have selected three network datasets, for which the community structure is already known from other sources. Table 1 shows the detected community number and the ARI score of discovered structures of both algorithms. The first network is drawn from the well-known "karate club" study of Zachary [44]. In this study, relations between 34

	Ground Truth	Q based Algorithm N		Our HMaxMin Algorithm		Improvement
	Communities	Comm.	ARI	Comm.	ARI	
Karate Club	2	3	0.680	2	1.00	47.1 %
Sawmill Network	3	4	0.664	3	1.00	50.6%
Mexican Politicians	2	3	0.255	3	0.359	40.7%

Table 1: Algorithm Comparison on Real World Networks.

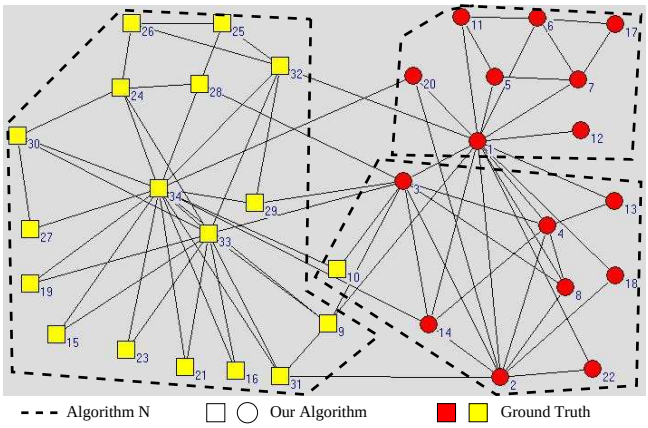

Figure 6: The Karate Club

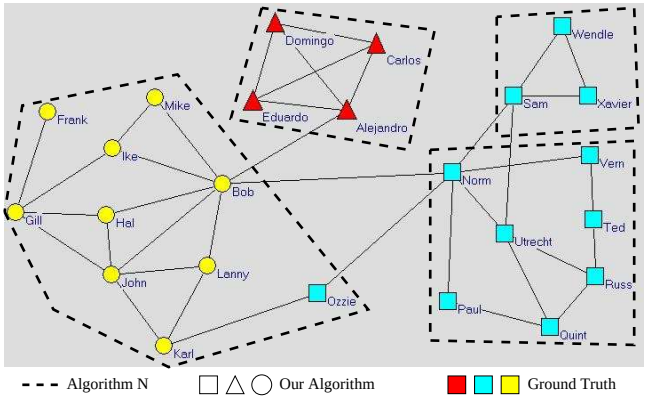

Figure 7: Social Network in a Sawmill

members of a karate club over a period of two years are observed. During the study, a disagreement developed between the administrator and the teacher of the club, which eventually made the club split into two smaller ones, centering around the administrator and the teacher, represented by node 34 and node 1. Then Zachary was able to construct a network of friendships, using a variety of measures to estimate the strength of ties between members of the club.

The user-defined criteria we used for the karate network and all the following experiments is a heuristic rule for social networks where we believe the neighbourhood around people are as important as direct relations, thus disconnected people are treated as *related pairs* if they share the same intermediary friends. In other words, this criteria only penalizes the MM modularity for pairs that are in the same community and have no shared friends. We reward the score for connected pairs and do not reward or penalize sharing-neighbour pairs.

In Figure 6[1], we show a unweighted network structure extracted from Zachary's observations. The actual divisions of the club following the break-up, as revealed by which club the members attended afterward, are indicated by node colours. We also show the results after feeding the network into HMaxMin (represented by node shape) and algorithm N (represented by closed area) in the figure. As we can see, algorithm N not only incorrectly generates one extra community, but also classifies node 10 into the wrong community[2], while results of our algorithm perfectly match the ground truth. In other words, the MM Modularity is better than the original as a predictor of subsequent social evolution of this friendship network.

5.5 Sawmill Communication Network As a further test of our algorithm, we turn to the communication network of employees in a sawmill[3]. This data is collected in order to analyze the communication structure among the employees after a strike. An edge in the network means that the two connected employees have discussed the strike with each other very often. As Figure 7 shows, there are three groups according to age and language. The Spanish-speaking young employees (the top group) are almost disconnected from the English-speaking young employees (the left group), who communicate with no more than two of the older English-speaking employees (the right group). All ties between groups have special backgrounds. For example, Alejandro is most proficient in English and Bob speaks some Spanish, which explains their connection. Bob

[1] All following network figures are generated using Pajek [31].

[2] The result in [27], which showed that algorithm N find two communities with only one misclassified node, is incorrect, confirmed by M. Newman in private communication.

[3] This dataset is collected from the Pajek Project [31].

owes Norm for getting his job, which may be the reason that they developed a friendship tie. Finally, Ozzie is Karl's father [31].

In Figure 7, we show the communities of the ground truth, our algorithm and algorithm N, indicated again by the node colour, shape and closed area. As we can see, both algorithms correctly identify the Spanish-speaking group, which is a clique. However, algorithm N inaccurately classifies Ozzie into the young English-speaker group and generates an extra community for the older English-speaker group although Sam strongly connects the two separated groups. On the other hand, HMaxMin again perfectly detects the ground truth, as revealed by the sociology research.

5.6 Mexican Politician Network For our next example, we look at a more complex relation network between politicians in Mexico (also collected from the Pajek Project [31]), which describes a social network between Mexican politicians in the 20th century. Edges represent significant social ties between the politicians, represented by nodes. Two groups within this network have been competing for power against each other (Figure 8), which are civilians (the top group) and members of the military force (the bottom group).

As we can see in Figure 8, this network has way more between-community connections than the previous two networks, which makes it harder for community mining algorithms to correctly detect the communities (as shown by the experiments on synthetic data). As the figure shows, algorithm N finds most members of the civilian group (red nodes), but it separates the military group (yellow nodes) into two communities and makes several mistakes around the periphery. On the other hand, HMaxMin also detects three communities, one for the civilian group, one for military group and one in the middle, mainly containing periphery nodes. Although it is hard to argue which partition is better simply by observation, Table 1 shows that the HMaxMin algorithm achieves a better 0.359 ARI score than 0.255 of algorithm N.

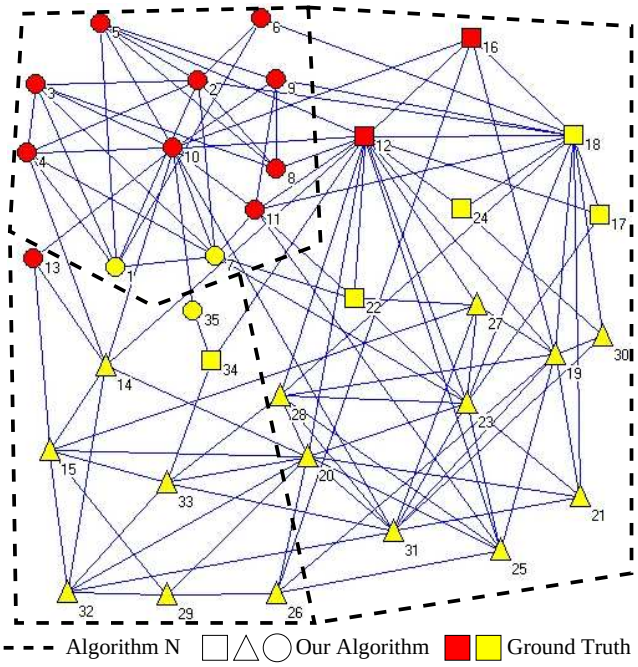

Figure 8: Mexican Politician Network

6 Discussion

Recently, community mining is increasingly attracting attention as an area of study and faces many challenges in developing community structures. Newman's Modularity has been proven to be effective and is thus pursued by many researchers, however, it has three main problems as we reviewed in Section 3. Our Max-Min modularity solves the third problem of modularity shown in Figure 1, which does not consider absent links, by including the factor of user-defined related node pairs in the quality measure process, thus not only the detection accuracy is improved by taking advantage of domain knowledge, but the community structure in different graphs can be compared. The Max-Min modularity idea can easily be extended to its local version, where global information about the graph is unavailable, and can be used in the recursive community detection to improve the community resolution. In networks such as biological and social networks, where connections can be unobserved, only considering the connected pairs might be inaccurate for community structure detection. While other algorithms cannot handle such cases, our MM modularity-based methods can exploit information from link prediction [23], and extract appropriate criteria for community detection. Additionally, our algorithm still runs in $O(mD \log n)$ time, which is the same as previous modularity-based algorithms.

7 Conclusions

We have described a new measure based on modularity for community structure and a hierarchical clustering algorithm HMaxMin for detecting communities from various networks. Different from other similar algorithms, which use one pre-defined similarity measure for all kinds of networks, our approach takes domain knowledge into consideration and thus improves the community detection accuracy. The proposed measure not only considers to maximize connected node pairs but also to minimize unrelated pairs in the same community, thus

provides a considerable improvement over the original modularity, which only measures the existing connections within communities. While giving a penalty for all absent links might be too strict, domain knowledge is incorporated in our model to boost performance. Our change on the modularity has a big impact on community detection, and further improves the high accuracy that modularity-based method already achieve. We have applied the algorithm to randomly generated networks and a set of real world networks with ground truth for validation. We have also applied the algorithm on large networks to show its scalability. The experimental results confirm the accuracy and effectiveness of the proposed measure and algorithm for community structure detection.

References

[1] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek. Robust, complete, and efficient correlation clustering. In *Proc. 7th SIAM International Conference on Data Mining (SDM'07)*, 2007.

[2] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *Machine Learning*, pages 238–247, 2002.

[3] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *ICML*, pages 27–34, 2002.

[4] S. Boccaletti, M. Ivanchenko, V. Latora, A. Pluchino, and A. Rapisarda. Detection of complex networks modularity by dynamical clustering. *Physical Review E*, 75:045102, 2007.

[5] M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. In *FOCS*, page 524, 2003.

[6] A. Clauset. Finding local community structure in networks. *Physical Review E*, 72:026132, 2005.

[7] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very lage networks. *Phys. Rev. E*, 70:066111, 2004.

[8] L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas. Comparing community structure identification. *J. Stat. Mech*, page P09008, 2005.

[9] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD*, pages 89–98, 2003.

[10] C. H. Q. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM*, pages 107–114, 2001.

[11] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72:027104, 2005.

[12] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *KDD*, pages 150–160, 2000.

[13] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *PROC.NATL.ACAD.SCI.USA*, 104:36, 2007.

[14] L. Getoor and C. P. Diehl. Link mining: a survey. *SIGKDD Explor. Newsl.*, 7(2):3–12, 2005.

[15] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. In *ICDE*, pages 341–352, 2005.

[16] I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. In *SODA*, pages 1167–1176, 2006.

[17] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. In *PNAS USA, 99:8271-8276*, 2002.

[18] S. Gregory. An algorithm to find overlapping community structure in networks. In *PKDD*, pages 91–102, 2007.

[19] R. Guimera and L. A. N. Amaral. Functional cartography of complex metabolic networks. *Nature*, 433:895–900, 2005.

[20] IMDB. http://www.imdb.com/.

[21] D. Jensen. Statistical challenges to inductive inference in linked data, 1999.

[22] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distriuted Computing*, 48(1):96–129, 1998.

[23] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM*, pages 556–559, 2003.

[24] B. Long, X. Wu, Z. M. Zhang, and P. S. Yu. Unsupervised learning on k-partite graphs. In *KDD*, pages 317–326, 2006.

[25] B. Long, Z. M. Zhang, and P. S. Yu. A probabilistic framework for relational clustering. In *KDD*, pages 470–479, 2007.

[26] T. Nepusz, A. Petroczi, L. Negyessy, and F. Bazso. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77, 2008.

[27] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69, 2004.

[28] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74, 2006.

[29] M. E. J. Newman. Modularity and community structure in networks. *PNAS USA*, 103, 2006.

[30] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69, 2004.

[31] Pajek. http://vlado.fmf.uni-lj.si/pub/networks/pajek/.

[32] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814, 2005.

[33] J. Ruan and W. Zhang. An efficient spectral algorithm for network community discovery and its applications to biological and social networks. In *ICDM*, pages 643–648, 2007.

[34] J. Ruan and W. Zhang. Identifying network communities with a high resolution. *Physical Review E*, 77:016104, 2008.

[35] J. Scripps, P.-N. Tan, and A.-H. Esfahanian. Exploration of link structure and community-based node roles in network. In *ICDM*, 2007.

[36] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE. Trans. on Pattern Analysis and Machine Intelligence*, 2000.

[37] C. Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *SODA*, pages 526–527, 2004.

[38] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *KDD*, pages 717–726, 2007.

[39] K. Wagstaff and C. Cardie. Clustering with instance-

level constraints. In *ICML*, pages 1103–1110, 2000.
[40] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, pages 577–584, 2001.
[41] S. White and P. Smyth. A spectral clustering approach to finding communities in graphs. In *SIAM*, 2005.
[42] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger. Scan: a structural clustering algorithm for networks. In *KDD*, pages 824–833, 2007.
[43] K. Y. Yip and M. K. Ng. Harp: A practical projected clustering algorithm. *IEEE TKDE*, 16(11):1387–1397, 2004.
[44] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.
[45] O. R. Zaïane, J. Chen, and R. Goebel. Dbconnect: Mining research community on dblp data. In *Joint 9th WEBKDD and 1st SNA-KDD Workshop*, 2007.
[46] S. Zhang, R.-S. Wang, and X.-S. Zhang. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A*, 374:483–490, 2007.