A General Probabilistic Framework for Detecting Community Structure in Networks

Cheng-Shang Chang, Chin-Yi Hsu, Jay Cheng, and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 30013, Taiwan, R.O.C.
Email: cschang@ee.nthu.edu.tw; g9764514@oz.nthu.edu.tw;
jcheng@ee.nthu.edu.tw; lds@cs.nthu.edu.tw

***Abstract*—Based on Newman's fast algorithm [13], in this paper we develop a general probabilistic framework for detecting community structure in a network. The key idea of our generalization is to characterize a network (graph) by a bivariate distribution that specifies the probability of the two vertices appearing at both ends of a randomly selected *path* in the graph. With such a bivariate distribution, we give a probabilistic definition of a community and a definition of a modularity index. To detect communities in a network, we propose a class of distribution-based clustering algorithms that have comparable computational complexity to that of Newman's fast algorithm. Our generalization provides the additional freedom to choose a bivariate distribution and a correlation measure. As such, we obtain significant performance improvement over the original Newman fast algorithm in the computer simulations of random graphs with known community structure.**

keywords: large complex networks, graph partitioning, clustering algorithms

I. Introduction

There has been a surge of interest on detecting community structure in large complex networks since the first paper in the physics literature by Girman and Newman [9]. In the literature, networks are commonly modelled by (mathematical) graphs and the problem of detecting community structure in large complex networks is also known as the *graph partitioning problem* that divides a graph into several disjoint subgraphs, called *clusters* or *communities*. As pointed out in the recent review papers in [11] and [20], many algorithms have been developed by researchers among various research communities, including physicists, biologists, and computer scientists. These algorithms might be classified as follows: (i) divisive algorithms [15], [21], [26], [6], [22], (ii) agglomerative algorithms [13], [2], (iii) graph partitioning and clustering algorithms [4], [16], and (iv) data compression algorithms [23], [24]. Various comparison studies of these algorithms can be found in [5], [17].

In spite of all the efforts in developing community detection algorithms, there are still many questions that we do not have satisfactory answers. For instance, what is a community in a network? Even with a definition of a community, what would be the right index for measuring the performance of a graph partition? Based on Newman's fast algorithm [13], in this paper we will provide a general probabilistic framework for these questions. The key idea of our framework is to characterize a graph by a bivariate distribution that specifies the probability of the two vertices appearing at both ends of a "randomly" selected *path* in the graph. With such a bivariate distribution, we can then define a community as a set of vertices with the property that it is more likely to find the other end in the same community given one of the two ends in a randomly selected path is already in the community. To detect communities, we define a class of correlation measures that can be used for measuring how two vertices (and two communities) are related. Two communities are positively (resp. negatively) correlated if the value of a correlation measure for these two communities is positive (resp. negative). As a generalization of Newman's fast algorithm, we propose a class of distribution-based clustering algorithms for community detection. Like most agglomerative algorithms, our distribution-based clustering algorithms start from viewing each vertex as a solely member in a community and then repeatedly merge the two most positively correlated communities into a new community until all the remaining communities are negatively correlated. There are two theoretic results that can be proved for a distribution-based clustering algorithm: (i) it guarantees that every community detected by the algorithm satisfies the definition of a community under certain technical conditions for the bivariate distribution, and (ii) the algorithm increases an index in each merge of two positively correlated communities. Such an index is also called a modularity index in this paper as it is a generalization of the original modularity index in [13] and it might be used as a performance index for a graph partition. Once the bivariate distribution is given, the computation complexity of a distribution-based clustering algorithm is $O(n^2 \log n)$ for a graph with n vertices and it can be reduced to $O(n(\log n)^2)$ following the implementation in [2] by exploiting the "sparseness" of the bivariate distribution.

One of the well-known problems of Newman's fast algorithm is its resolution limit in detecting communities [12]. Our general probabilistic framework might provide a solution for this problem. Note that for each choice of the bivariate distribution there is one corresponding definition of a community and one corresponding definition of a modularity index.

Newman's fast algorithm simply corresponds to the special case that the bivariate distribution is obtained from *uniformly* selecting a path with length 1. As such, its resolution is quite limited. To improve the resolution, it seems plausible to consider bivariate distributions that have nonzero probabilities for selecting paths with length greater than 1. Such an observation is verified via extensive computer simulations for randomly generated graphs with 128 vertices and four known communities in this paper.

In addition to the choice of the bivariate distribution for the problem of resolution limit, there is another choice of the correlation measure that might lead to performance improvement. In this paper, we propose three correlation measures: (i) covariance, (ii) correlation and (iii) mutual information. The first one corresponds to the original measure used in Newman's fast algorithm. From our simulation results, the last two perform better than the first one when we consider a bivariate distribution that has nonzero probabilities for selecting paths with length greater than 1.

The rest of the paper is organized as follows. In Section II, we first give a brief review of Newman's fast algorithm. We then provide a probabilistic interpretation of Newman's fast algorithm in Section III. The general framework is given in details in Section IV. We show how one can obtain a bivariate distribution from the adjacency matrix of a graph in Section IV-A, define correlation measures in Section IV-B, propose the class of distribution-based clustering algorithms in Section IV-C, and define a community and a modularity index in Section IV-D. We report our simulation results in Section V. The paper is concluded in Section VI, where we address possible extensions of our work.

II. Review of Newman's fast algorithm

In the literature, a network is commonly modelled by a graph $G(V, E)$, where V denotes the set of vertices in the graph and E denotes the set of edges in the graph. The problem of detecting community structure in a network is to find a function that assigns every vertex in the graph to a community (also known as a graph partitioning problem in [7], [25]). In this paper, we are particularly interested in Newman's fast algorithm [13] for finding such an assignment, and we will start from giving a brief review of Newman's fast algorithm.

Let $n = |V|$ be the number of vertices in the graph and index the n vertices from $1, 2, \ldots, n$. Then the graph $G(V, E)$ can also be characterized by an $n \times n$ adjacency matrix A, where

$$A_{vw} = \begin{cases} 1, & \text{if vertices } v \text{ and } w \text{ are connected,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Let $m = |E|$ be the number of edges in the graph and k_v be the degree of vertex v. From the adjacency matrix, we then have

$$m = \frac{1}{2} \sum_{v=1}^{n} \sum_{w=1}^{n} A_{vw}, \tag{2}$$

and

$$k_v = \sum_{w=1}^{n} A_{vw}. \tag{3}$$

Let c_v be the community of vertex v and $\delta(c_v, i)$ be the δ-function that equals to 1 if $c_v = i$ and 0 otherwise. Then the fraction of ends of edges that are attached to the vertices in community i, denoted by a_i, can be represented as follows:

$$a_i = \frac{1}{2m} \sum_{v=1}^{n} k_v \delta(c_v, i). \tag{4}$$

Let

$$e_{ij} = \frac{1}{2m} \sum_{v=1}^{n} \sum_{w=1}^{n} A_{vw} \delta(c_v, i) \delta(c_w, j). \tag{5}$$

When $i = j$, e_{ij} is the fraction of edges that join the vertices in community i, and when $i \neq j$, e_{ij} is one-half of the fraction of edges that join the vertices in community i and the vertices in community j.

In [15], Newman and Girvan proposed a modularity index Q as follows:

$$Q = \sum_{i} (e_{ii} - a_i^2). \tag{6}$$

As explained in [13], if the fraction of within-community edges is the same as what we would expect for a randomized network, then this quantity is zero. Nonzero values represent deviations from randomness. The objective of a community-detecting algorithm is then to find an assignment for each vertex so that the modularity index Q can be maximized. However, it was shown in [1] that finding such an optimal assignment is NP-complete in the strong sense.

In [13], Newman proposed a heuristic approach for the problem based on an agglomerative hierarchical clustering method (see e.g., the books [7], [25] for more references on agglomerative hierarchical clustering algorithms). The algorithm starts with a state in which each vertex is the sole member in its community. Then one repeatedly joins communities together in pairs by choosing at each step the join that results in the greatest increase (or smallest decrease) in the modularity index Q. To see how the algorithm works, suppose that there are C communities in a certain step with

$$\begin{aligned} Q = (e_{11} - a_1^2) + \cdots + (e_{ii} - a_i^2) + \cdots \\ + (e_{jj} - a_j^2) + \cdots + (e_{CC} - a_C^2). \end{aligned}$$

Now suppose we group community i and community j to form a new community k. As e_{kk} is the fraction of edges that joins the vertices in communities i and j and a_k is the fraction of ends of edges that are attached to the vertices in communities i and j, it is easy to see that $e_{kk} = e_{ii} + 2e_{ij} + e_{jj}$ and $a_k = a_i + a_j$.

Thus, the modularity index Q after grouping community i and community j to form a new community k is

$$Q = (e_{11} - a_1^2) + \cdots + (e_{kk} - a_k^2) + \cdots + (e_{CC} - a_C^2),$$

where

$$(e_{kk} - a_k^2) = (e_{ii} + e_{jj} + 2e_{ij}) - (a_i + a_j)^2 \\ = (e_{ii} - a_i^2) + (e_{jj} - a_j^2) + 2(e_{ij} - a_i a_j).$$

This shows that the change of the modularity index, denoted by ΔQ_{ij}, can be easily computed as follows:

$$\Delta Q_{ij} = 2(e_{ij} - a_i a_j). \tag{7}$$

Newman's fast algorithm then chooses the pair of community i and community j and group them to form a new community so that ΔQ_{ij} is maximized in each step.

III. A probabilistic interpretation of Newman's fast algorithm

To further understand the intuition behind Newman's fast algorithm, we provide a probabilistic interpretation for Newman's fast algorithm in this section.

Consider the graph $G(V, E)$ as described in the previous section. Suppose that an edge is selected *uniformly* from the m edges of the graph. Let (V, W) be the bivariate random vector that represents the vertices at the two ends of the random selected edge. Then we have

$$\mathsf{P}((V, W) = (v, w)) \\ = \begin{cases} \frac{1}{2m}, & \text{if vertices } v \text{ and } w \text{ are connected,} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

From (8), it follows that

$$\mathsf{P}(V = v) = \sum_{w=1}^{n} \mathsf{P}((V, W) = (v, w)) = \frac{k_v}{2m}, \tag{9}$$

where k_v is the degree of vertex v. Similarly,

$$\mathsf{P}(W = w) = \sum_{v=1}^{n} \mathsf{P}((V, W) = (v, w)) = \frac{k_w}{2m}. \tag{10}$$

Let S_i be the set of nodes in community i and X_i (resp. Y_j) be the indicator variable for the event that V is in community i (resp. W is in community j). Specifically,

$$X_i = \begin{cases} 1, & V \in S_i, \\ 0, & V \notin S_i, \end{cases} \tag{11}$$

and

$$Y_j = \begin{cases} 1, & W \in S_j, \\ 0, & W \notin S_j. \end{cases} \tag{12}$$

Using (9) yields

$$\mathsf{P}(X_i = 1) = \mathsf{P}(V \in S_i) = \sum_{v \in S_i} \frac{k_v}{2m} \\ = \frac{1}{2m} \sum_{v} k_v \delta(c_v, i) = a_i, \tag{13}$$

where we use (4) in the last identity. Similarly, we also have

$$\mathsf{P}(Y_j = 1) = \mathsf{P}(W \in S_j) = a_j, \tag{14}$$

and

$$\mathsf{P}(X_i = 1, Y_j = 1) = \mathsf{P}(V \in S_i, W \in S_j) \\ = \frac{1}{2m} \sum_{v=1}^{n} \sum_{w=1}^{n} A_{vw} \delta(c_v, i) \delta(c_w, j) = e_{ij}. \tag{15}$$

Using (13)-(15), one can then compute the covariance of X_i and Y_i as follows:

$$\mathrm{Cov}(X_i, Y_j) = \mathsf{E}[X_i Y_j] - \mathsf{E}[X_i]\mathsf{E}[Y_j] \\ = \mathsf{P}(X_i = 1, Y_j = 1) - \mathsf{P}(X_i = 1)\mathsf{P}(Y_j = 1) \\ = e_{ij} - a_i a_j. \tag{16}$$

The covariance of two random variables is commonly used in the literature to measure how these two random variables are related. A positive (resp. negative) covariance indicates that these two random variables are *positively (resp. negatively) correlated*. For two indicator random variables (like those in (16)), they are *independent* if and only if their covariance is zero.

In view of (16) and (7), we have a very intuitive probabilistic interpretation for Newman's fast algorithm. Basically, one characterizes how two communities are related by a covariance matrix obtained by using (16) for all pairs of two communities. In each step, the two communities that has the largest covariance are selected and grouped into a new community. The covariance matrix is then updated. The process is repeated until either there is only one community left or all the remaining pairs of communities are negatively correlated.

IV. A general probabilistic framework

The probabilistic interpretation of Newman's fast algorithm inspires us to develop a general probabilistic framework for detecting community structure in a network. Instead of characterizing a network by a graph, we characterize a network by a bivariate distribution. Specifically, for a network with the set of nodes $\{1, 2, \ldots, n\}$, it is characterized by a bivariate distribution $p(v, w)$, $v, w = 1, 2, \ldots, n$, for randomly selecting a pair of two nodes V and W, i.e.,

$$\mathsf{P}(V = v, W = w) = p(v, w). \tag{17}$$

Let $p_V(v)$ (resp. $p_W(w)$) be the marginal distribution of the random variable V (resp. W), i.e.,

$$p_V(v) = \sum_{w=1}^{n} p(v, w), \tag{18}$$

and

$$p_W(w) = \sum_{v=1}^{n} p(v, w). \tag{19}$$

If $p(v, w)$ is *symmetric*, then $p_V(v) = p_W(v)$ for all v, and $p_V(v)$ is the probability that a randomly selected node is v. In case that $p(v, w)$ is not symmetric, we can find a symmetric bivariate distribution $\tilde{p}(v, w)$ by letting

$$\tilde{p}(v, w) = \frac{1}{2}(p(v, w) + p(w, v)).$$

A. From a graph to a bivariate distribution

As mentioned before, a network is commonly modelled by a graph $G(V, E)$, which in turn is characterized by an adjacency matrix A. The question is how one obtains a bivariate distribution characterization from a graph model. A direct approach is to follow the probabilistic interpretation in Section III that maps an adjacency matrix A to a bivariate distribution in (8). Let $\alpha(A)$ be the sum of all the elements in a matrix A, i.e.,

$$\alpha(A) = \sum_v \sum_w A_{vw}. \tag{20}$$

Then one can rewrite (8) as follows:

$$\mathsf{P}(V = v, W = w) = \frac{1}{\alpha(A)} A_{vw}. \tag{21}$$

One problem of using the adjacency matrix is the resolution limit in community detection. As argued in [12], optimizing the modularity index Q in [15] may fail to detect communities smaller than a scale that depends on the size of the network and the degree of interconnectedness of the communities. This motivates us to consider a more general approach that maps a graph to a bivariate distribution.

Recall that the bivariate distribution in (21) is the probability for the two ends of a randomly selected edge in a graph. Our idea is to generate the needed bivariate distribution by randomly selecting the two ends of a *path*. For this, we first consider a (matrix) function f that maps an adjacency matrix A to another matrix $f(A)$. Then we define a bivariate distribution from $f(A)$ by

$$\mathsf{P}(V = v, W = w) = \frac{1}{\alpha(f(A))} f(A)_{vw}. \tag{22}$$

This idea is further illustrated in the following example for randomly selecting two ends of a path with length not greater than 2.

Example 1: **(A random selection of a path with length not greater than 2)** Consider a graph with an $n \times n$ adjacency matrix A and

$$f(A) = \lambda_0 \mathbf{I} + \lambda_1 A + \lambda_2 A^2, \tag{23}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix, and λ_0, λ_1, and λ_2 are three *nonnegative* constants. Then the two random variables V and W in (22) represents the two ends of a randomly selected path with length not greater than 2. To see this, note that there are n paths with length 0 (for the n vertices), $\alpha(A)$ paths with length 1, and $\alpha(A^2)$ paths with length 2. Since

$$\alpha(f(A)) = \lambda_0 \alpha(\mathbf{I}) + \lambda_1 \alpha(A) + \lambda_2 \alpha(A^2),$$

a path with length ℓ is selected with probability $\lambda_\ell / \alpha(f(A))$ for $\ell = 0, 1$, and 2.

We note that the computation complexity of A^2 is $O(mn)$ for a sparse $n \times n$ matrix with at most m nonzero elements and there exist other fast sparse matrix multiplication algorithms in the literature (see e.g., [27]).

Another approach to generate a bivariate distribution from the adjacency matrix A from a graph is to consider a random walk on a graph (see e.g., [3]).

Example 2: **(A random walk on a graph)** Consider a graph with an $n \times n$ adjacency matrix A. As in (2) and (3), let m be the total number of edges and k_v be the degree of vertex v. A random walk on such a graph can be characterized by a Markov chain with the $n \times n$ transition probability matrix $R = (R_{v,w})$, where

$$R_{v,w} = \frac{1}{k_v} A_{vw} \tag{24}$$

is the transition probability from vertex v to vertex w. The stationary probability that the Markov chain is in vertex v, denoted by π_v, is $k_v / 2m$. Let β_ℓ be the probability that we select a path with length ℓ, $\ell = 1, 2, \ldots$. Then the probability of selecting a random walk (path) with vertices $v = v_1, v_2, \ldots, v_{\ell+1} = w$ is

$$\beta_\ell \pi_{v_1} \prod_{i=1}^{\ell} R_{v_i, v_{i+1}}. \tag{25}$$

From this, we then have the bivariate distribution

$$p(v, w) = \pi_v \sum_{\ell=1}^{\infty} \beta_\ell \sum_{v_2} \cdots \sum_{v_\ell} \prod_{i=1}^{\ell} R_{v_i, v_{i+1}}. \tag{26}$$

Since A is a *symmetric* matrix, it is easy to see that

$$\pi_v R_{v,w} = \frac{1}{2m} A_{v,w} = \frac{1}{2m} A_{w,v} = \pi_w R_{w,v}$$

for all v and w, and the Markov chain is thus a *reversible* Markov chain [19]. This implies that

$$\pi_{v_1} \prod_{i=1}^{\ell} R_{v_i, v_{i+1}} = \pi_{v_\ell} \prod_{i=1}^{\ell} R_{v_{i+1}, v_i}$$

and $p(v, w) = p(w, v)$ is thus a symmetric bivariate distribution. To randomly select a path with length not greater than 2, we can simply let $\beta_\ell = 0$ for all $\ell > 2$ and this leads to

$$p(v, w) = \frac{\beta_1}{2m} A_{v,w} + \frac{\beta_2}{2m} \sum_{v_2=1}^{n} \frac{A_{v,v_2} A_{v_2,w}}{k_{v_2}}. \tag{27}$$

B. Correlation measures

As discussed in Section III, Newman's fast algorithm uses *covariance* to measure how positively two indicator random variables are related. In this section, we extend this to a more general setting by considering "correlation measures" defined below.

Definition 3: For any two indicator random variables X and Y, $\rho(X, Y)$ is called a *correlation measure* in this paper if

- (C0) $\rho(X, Y)$ is solely determined by the bivariate distribution of X and Y,
- (C1) $\rho(X, Y) = 0$ if and only if X and Y are independent, i.e.,

$$\mathsf{P}(X = 1, Y = 1) = \mathsf{P}(X = 1)\mathsf{P}(Y = 1), \tag{28}$$

and

(C2) $\rho(X,Y) > 0$ if and only if X and Y are positively correlated, i.e.,

$$\mathsf{P}(X=1,Y=1) > \mathsf{P}(X=1)\mathsf{P}(Y=1). \quad (29)$$

From (C1) and (C2), we also know that $\rho(X,Y) < 0$ if and only if X and Y are negatively correlated, i.e.,

$$\mathsf{P}(X=1,Y=1) < \mathsf{P}(X=1)\mathsf{P}(Y=1). \quad (30)$$

We note that the bivariate distribution of two indicator random variables X and Y is determined once $\mathsf{P}(X=1,Y=1)$, $\mathsf{P}(X=1)$ and $\mathsf{P}(Y=1)$ are given. As such, we only need to store $\mathsf{P}(X=1,Y=1)$, $\mathsf{P}(X=1)$ and $\mathsf{P}(Y=1)$ in memory for the algorithms developed in the next section.

Example 4: **(Covariance)** For two indicator random variables X and Y, we have

$$\mathrm{Cov}(X,Y) = \mathsf{P}(X=1,Y=1) - \mathsf{P}(X=1)\mathsf{P}(Y=1).$$

Clearly, the covariance of X and Y is a correlation measure.

Example 5: **(Correlation)** Note that the correlation of two random variables X and Y, denoted by $\mathrm{Correl}[X,Y]$, can be computed as follows:

$$\mathrm{Correl}[X,Y] = \frac{\mathrm{Cov}[X,Y]}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}}, \quad (31)$$

where $\mathrm{Var}(X)$ (resp. $\mathrm{Var}(Y)$) is the variance of X (resp. Y). For two indicator random variables X and Y, we then have

$$\mathrm{Var}(X) = \mathsf{P}(X=1) - (\mathsf{P}(X=1))^2, \quad (32)$$

$$\mathrm{Var}(Y) = \mathsf{P}(Y=1) - (\mathsf{P}(Y=1))^2, \quad (33)$$

$$\mathrm{Correl}[X,Y] = \frac{\mathsf{P}(X=1,Y=1) - \mathsf{P}(X=1)\mathsf{P}(Y=1)}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}}. \quad (34)$$

Clearly, the correlation of X and Y is also a correlation measure. In comparison with the covariance of X and Y in Example 4, the correlation of X and Y always has a value between -1 and 1, and could be more suitable for our computation later.

Example 6: **(Mutual information)** The mutual information of two random variables X and Y (see e.g., [3]), denoted by $I(X;Y)$, can be computed as follows:

$$I(X;Y) = \sum_{x,y} P_{X,Y}(x,y) \log \frac{P_{X,Y}(x,y)}{P_X(x)P_Y(y)}, \quad (35)$$

where $P_{X,Y}(x,y)$ is the bivariate distribution of X and Y, $P_X(x)$ is the marginal distribution of X, and $P_Y(y)$ is the marginal distribution of Y. The mutual information $I(X;Y)$, also known as a special case of the Kullback-Leibler distance [3], is commonly used as a measure for the distance between the bivariate distribution $P_{X,Y}(x,y)$ of two random variables X and Y and the bivariate distribution if they were independent. When X and Y are independent, we have $P_{X,Y}(x,y) = P_X(x)P_Y(y)$. Thus $I(X;Y) = 0$ if and only if X and Y are independent. Let $\mathrm{Sgn}(x)$ be the sign function, i.e., $\mathrm{Sgn}(x) = 1$ if $x \geq 0$ and $\mathrm{Sgn}(x) = -1$ if $x < 0$. Then it is easy to verify that

$$\rho(X,Y) = \mathrm{Sgn}(\mathrm{Cov}(X,Y)) \cdot I(X;Y) \quad (36)$$

is also a correlation measure.

C. Distribution-based clustering algorithms

In the following, we propose a class of distribution-based clustering algorithms. Like Newmans's fast algorithm [13], our algorithms also fall in the category of agglomerative hierarchical clustering algorithms (see e.g., [7], [25]).

Distribution-based clustering algorithms:

(P0) Input a bivariate distribution $p(v,w)$, $v,w = 1,2,\ldots,n$ that characterizes the two randomly selected nodes V and W, and a correlation measure $\rho(X,Y)$ for two indicator random variables.

(P1) Initially, there are n communities, indexed from 1 to n, with each community containing exactly one node. Specifically, let S_i be the set of nodes in community i. Then $S_i = \{i\}$, $i = 1,2,\ldots,n$.

(P2) Let X_i (resp. Y_j) be the indicator random variable for the event that V is in community i (resp. W is in community j). Then

$$\mathsf{P}(X_i=1) = \sum_{v\in S_i} p_V(v) = p_V(i),$$

$$\mathsf{P}(Y_j=1) = \sum_{w\in S_j} p_W(w) = p_W(j),$$

$$\mathsf{P}(X_i=1,Y_j=1) = \sum_{v\in S_i, w\in S_j} p(v,w) = p(i,j).$$

Compute $\rho(X_i,Y_j)$ for all i and j.

(P3) Find the two (distinct) communities that have the largest correlation measure. Group these two communities into a new community. Suppose that community i and community j are grouped into a new community k. Then $S_k = S_i \cup S_j$ and update

$$\mathsf{P}(X_k=1) = \mathsf{P}(X_i=1) + \mathsf{P}(X_j=1), \quad (37)$$

$$\mathsf{P}(Y_k=1) = \mathsf{P}(Y_i=1) + \mathsf{P}(Y_j=1), \quad (38)$$

$$\mathsf{P}(X_k=1,Y_k=1) = \mathsf{P}(X_i=1,Y_i=1) + \mathsf{P}(X_i=1,Y_j=1) + \mathsf{P}(X_j=1,Y_i=1) + \mathsf{P}(X_j=1,Y_j=1). \quad (39)$$

Moreover, for all $\ell \neq k$, update

$$\mathsf{P}(X_k=1,Y_\ell=1) = \mathsf{P}(X_i=1,Y_\ell=1) + \mathsf{P}(X_j=1,Y_\ell=1), \quad (40)$$

$$\mathsf{P}(X_\ell=1,Y_k=1) = \mathsf{P}(X_\ell=1,Y_i=1) + \mathsf{P}(X_\ell=1,Y_j=1). \quad (41)$$

(P4) For all $\ell \neq k$, compute $\rho(X_k,Y_\ell)$ and $\rho(X_\ell,Y_k)$.

(P5) Repeat (P3) until either there is only one community left or all the remaining pairs of communities have negative correlation measures, i.e., $\rho(X_i, Y_j) < 0$ for all $i \neq j$.

Distribution-based clustering algorithms are generalizations of Newmans's fast algorithm. In each iteration, the distribution is updated and then used for computing the new correlation measures. This is different from most *distance-based* clustering algorithms [7], [25], where the new distance between clusters is updated *directly*. Note that there are at most $n-1$ iterations in the above algorithm and there are $O(n)$ updates for the measures in (P4) for each iteration. The hard part is to find the two communities that have the largest correlation measure in (P3). If we simply use a linear search to find the two communities that have the largest correlation measure in each iteration, then its computational complexity is $O(n^2)$ and the overall computational complexity for the above algorithm is $O(n^3)$. To reduce the computational complexity, one can implement a sorted list for the measures in (P2) and then insert every measure update into the sorted list. As each insertion of a new update takes $O(\log(n))$ steps (by using a binary search) and there are $O(n)$ updates in each iteration, the computational complexity in each iteration can be reduced to $O(n\log(n))$ and that yields $O(n^2\log(n))$ computational complexity for the above algorithm. One can further reduce the computational complexity by exploring the "spareness" of the bivariate distribution. Suppose that we stop the algorithm in (P5) once all the remaining pairs of communities do not have *positive* measures, i.e., $\rho(X_i, Y_j) \leq 0$ for all i and j. In this case, we only need to maintain the list of measures with positive values. From (C2) in Definition 3, it suffices to maintain the pair of communities i and j with $P(X_i = 1, Y_j = 1) > 0$. Two communities i and j are said to be *connected* if either $P(X_i = 1, Y_j = 1) > 0$ or $P(X_j = 1, Y_i = 1) > 0$. Suppose that there are only $O(m)$ connected pairs of them at the beginning. In view of (P2), we only need to maintain (and update) the pair of *connected* communities. As such, instead of having $O(n)$ updates, one only needs $O(|i| + |j|)$ updates in each iteration, where $|i|$ (resp. $|j|$) is the number of communities connected to community i (resp. $|j|$). Thus, each iteration takes $O((|i|+|j|)\log n)$ steps. Analogous to the argument in [2], each connected pair contributes at most $2d$ updates till the end of the algorithm, where d the is depth of the dendrogram. Thus, the overall computational complexity for the above algorithm is $O(md\log n)$, In practice, we often have $m = O(n)$ and $d = O(\log n)$ and the computational complexity of the distribution-based clustering algorithm is $O(n(\log n)^2)$ as in [2].

D. A probabilistic definition of a community

Up to this point, we have not defined what a community means. In the literature, there are many definitions for communities based the adjacency matrix of the graph that characterizes a network (see e.g., [21], [10]).

Here we provide a probabilistic definition of a community based on our framework.

Definition 7: A set of nodes S is a *community* in a probabilistic sense if

$$\mathsf{P}(V \in S, W \in S) \geq \mathsf{P}(V \in S)\mathsf{P}(W \in S). \tag{42}$$

If $\mathsf{P}(W \in S) > 0$, then this is equivalent to

$$\mathsf{P}(V \in S | W \in S) \geq \mathsf{P}(V \in S). \tag{43}$$

For a symmetric bivariate distribution $p(v, w)$, $\mathsf{P}(V \in S)$ is simply the probability that a randomly selected node is in the community. In comparison with the event that a randomly selected node is in the community, it is more likely to find the other node in the same community given that one of a randomly selected pair of two nodes is already in the community.

Analogous to the definition of the modularity index Q in [15], we define a modularity index based on our probabilistic framework.

Definition 8: Consider a bivariate distribution $p(v, w)$ with $v, w = 1, 2, \ldots, n$. Let S_c, $c = 1, 2, \ldots, C$, be a partition of $\{1, 2, \ldots, n\}$, i.e., $S_c \cap S_{c'}$ is an empty set for $c \neq c'$ and $\cup_{c=1}^{C} S_c = \{1, 2, \ldots, n\}$. The modularity index Q with respect to the partition S_c, $c = 1, 2, \ldots, C$, is

$$\sum_{c=1}^{C} \Big(\mathsf{P}(V \in S_c, W \in S_c) - \mathsf{P}(V \in S_c)\mathsf{P}(W \in S_c) \Big). \tag{44}$$

In the following theorem, we show (under certain technical conditions) that the modularity index is non-decreasing in every iteration of any distribution-based clustering algorithm and it indeed detects communities in the probabilistic sense defined in Definition 7.

Theorem 9: Suppose that $p(v, w)$ is symmetric.

(i) Then for any distribution-based clustering algorithm described in Section IV-C, the modularity index is non-decreasing in every iteration.

(ii) If, furthermore,

$$p(v, v) \geq (p_V(v))^2, \tag{45}$$

for all $v = 1, 2, \ldots, n$, then every community detected by any distribution-based clustering algorithm described in Section IV-C is a community in the probabilistic sense defined in Definition 7.

Proof. (i) Since we assume that $p(v, w)$ is symmetric, it suffices to show that

$$\sum_{c=1}^{C} \Big(\mathsf{P}(V \in S_c, W \in S_c) - (\mathsf{P}(V \in S_c))^2 \Big) \tag{46}$$

is non-decreasing in every iteration. Suppose that community i and community j are selected and grouped into a new community k in some iteration. Thus, we have a new partition of $\{1, 2, \ldots, n\}$ with $S_k = S_i \cup S_j$. To prove that the modularity index in (46) is non-decreasing after grouping

community i and community j into a new community k, we need to show that

$$\begin{aligned}&\mathsf{P}(V \in S_k, W \in S_k) - (\mathsf{P}(V \in S_k))^2 \\ &\geq \mathsf{P}(V \in S_i, W \in S_i) - (\mathsf{P}(V \in S_i))^2 \\ &\quad + \mathsf{P}(V \in S_j, W \in S_j) - (\mathsf{P}(V \in S_j))^2. \end{aligned} \tag{47}$$

Since $S_k = S_i \cup S_j$ and $S_i \cap S_j$ is an empty set, we have

$$\begin{aligned}&\mathsf{P}(V \in S_k, W \in S_k) \\ &= \mathsf{P}(V \in S_i, W \in S_i) + \mathsf{P}(V \in S_i, W \in S_j) \\ &\quad + \mathsf{P}(V \in S_j, W \in S_i) + \mathsf{P}(V \in S_j, W \in S_j), \end{aligned} \tag{48}$$

and

$$\mathsf{P}(V \in S_k) = \mathsf{P}(V \in S_i) + \mathsf{P}(V \in S_j). \tag{49}$$

Since community i and community j are selected, we know from (P5) in the distribution-based clustering algorithm that $\rho(X_i, Y_j) \geq 0$. In view of (C1) and (C2) in Definition 3, we have

$$\begin{aligned}&\mathsf{P}(V \in S_i, W \in S_j) = \mathsf{P}(X_i = 1, Y_j = 1) \\ &\geq \mathsf{P}(X_i = 1)\mathsf{P}(Y_j = 1) = \mathsf{P}(V \in S_i)\mathsf{P}(W \in S_j).\end{aligned}$$

From the symmetric property of $p(v, w)$, it follows that

$$\begin{aligned}&\mathsf{P}(V \in S_j, W \in S_i) = \mathsf{P}(V \in S_i, W \in S_j) \\ &\geq \mathsf{P}(V \in S_i)\mathsf{P}(V \in S_j). \end{aligned} \tag{50}$$

In view of (48), (49) and (50), it is straightforward to verify the inequality in (47).

(ii) We prove this by induction. Initially, we have $S_i = \{i\}$. It then follows from the assumption in (45) and the symmetric property of $p(v, w)$ that

$$\begin{aligned}&\mathsf{P}(V \in S_i, W \in S_i) = \mathsf{P}(V = i, W = i) = p(i, i) \\ &\geq (p_V(i))^2 = p_V(i)p_W(i) = \mathsf{P}(V \in S_i)\mathsf{P}(W \in S_i). \end{aligned} \tag{51}$$

In view of (51), we know all the initial n communities are communities in the probabilistic sense defined in Definition 7.

As the induction hypothesis, suppose that all the communities are communities in the probabilistic sense defined in Definition 7 up to the t^{th} iteration. Now suppose that community i and community j are selected and grouped into a new community k at the $(t+1)^{th}$ iteration. Thus, we have $S_k = S_i \cup S_j$. From the induction hypothesis, we know that

$$\begin{aligned}&\mathsf{P}(V \in S_i, W \in S_i) \geq \mathsf{P}(V \in S_i)\mathsf{P}(W \in S_i), \\ &\mathsf{P}(V \in S_j, W \in S_j) \geq \mathsf{P}(V \in S_j)\mathsf{P}(W \in S_j).\end{aligned}$$

It then follows from the symmetric property of $p(v, w)$ and (47) that

$$\begin{aligned}&\mathsf{P}(V \in S_k, W \in S_k) \geq (\mathsf{P}(V \in S_k))^2 \\ &= \mathsf{P}(V \in S_k)\mathsf{P}(W \in S_k). \end{aligned} \tag{52}$$

Thus, community k is also a community in the probabilistic sense defined in Definition 7. ■

We note that the condition in (45) is not satisfied for the choice of the bivariate distribution in (8). Thus, Newman's fast algorithm may not guarantee that every detected community is a community in the probabilistic sense defined in Definition 7.

In (P5), when $\rho(X_i, Y_j) < 0$ for all the remaining pairs of communities $i \neq j$, we know from Theorem 9(i) that the modularity index Q in Definition 8 cannot be further improved and this yields the best structure (in terms of the modularity index Q) by any distribution-based clustering algorithm (for a symmetric bivariate distribution). In practice, one might continue the process until there is only one community. In that case, a distribution-based clustering algorithm generates a dendrogram (an ordered binary tree) that specifies the order for two communities to be grouped into a new one in each step.

V. Simulation results

In this section, we report our simulation results. We will consider three distribution-based clustering algorithms: (i) covariance algorithm with the correlation measure in Example 4, (ii) correlation algorithm with the correlation measure in Example 5, and (iii) mutual information algorithm with the correlation measure in Example 6. To map a graph with an adjacency matrix A to a bivariate distribution $p(v, w)$, we also consider the following three types of functions in (23) of Example 1: (i) $(\lambda_0, \lambda_1, \lambda_2) = (0, 1, 0)$, i.e., $f_1(A) = A$, (ii)$(\lambda_0, \lambda_1, \lambda_2) = (1, 1, 0)$, i.e., $f_2(A) = \mathbf{I} + A$ and (ii)$(\lambda_0, \lambda_1, \lambda_2) = (1, 0.5, 0.25)$, i.e., $f_3(A) = \mathbf{I} + 0.5A + 0.25A^2$.

A. Random graphs with known community structure

In this section, we report our simulation results for a large number of random graphs with known community structure. For each random graph, there are $n = 128$ vertices and they are divided into four *known* communities with 32 vertices each (as in [13]). Let z_{in} be the parameter that represents the average number of edges from a vertex to the other vertices in the *same* community. Also, let z_{out} be the parameter that represents the average number of edges from a vertex to the other vertices in *different* communities. Let k be the average degree of a vertex. Then $k = z_{in} + z_{out}$. In our simulations, we fix $k = 16$ and vary z_{out} from 0 to 8. Once k and z_{out} are chosen, we generate the Erdös-Rényi [8] types of random graphs by connecting two vertices within the same communities with probability p_{in} and two vertices in different communities with probability p_{out}. We choose the probabilities $p_{in} = \frac{z_{in}}{\frac{n}{4}-1}$ and $p_{out} = \frac{z_{out}}{\frac{3n}{4}}$ so that the the average number of edges from a vertex to the other vertices in the *same* community is z_{in} and the the average number of edges from a vertex to the other vertices in *different* communities is z_{out}.

For each generated random graph, we detect the community structure by using the three algorithms and the three types of bivariate distributions described at the beginning of Section V. In our experiments for random graphs, we run

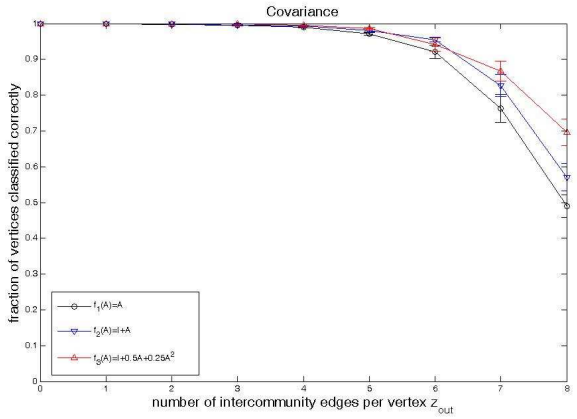

Fig. 1. Performance of $f_1(A)$, $f_2(A)$ and $f_3(A)$ under the covariance algorithm

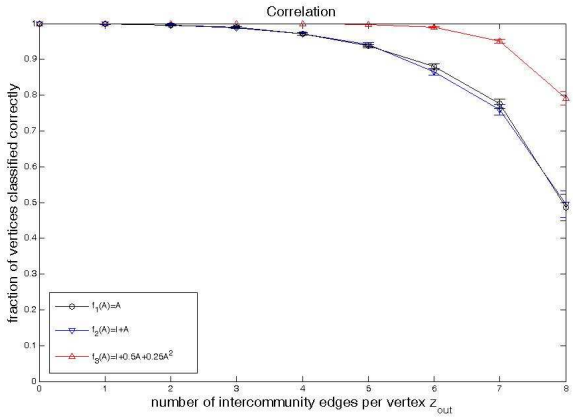

Fig. 2. Performance of $f_1(A)$, $f_2(A)$ and $f_3(A)$ under the correlation algorithm

the distribution-based clustering algorithms until there are exactly four communities left. To evaluate the performance of these algorithms, we compute the percentage of nodes that are correctly assigned. For this, we first identify the largest subset of vertices that are assigned to each of the four known communities. If two or more of these subsets belong to the same community, then all vertices in these subsets are considered incorrectly classified. Otherwise, the vertices in the four subsets are considered correctly classified. In Figure 1 (resp. Figure 2, Figure 3), we show the percentage of nodes that are correctly classified under the covariance (resp. correlation, mutual information) algorithm as a function of z_{out}. Each point in these figures is an average over 100 random graphs. In these figures, we also show 95% confidence intervals for all data points. From these three figures, it is clear that the choice of using $f_3(A) = \mathbf{I} + 0.5A + 0.25A^2$ significantly outperforms the other two choices, especially when z_{out} is large. The intuition behind this can be explained by considering the illustrating example in Figure 4. In the figure, there are two clearly separated communities A and B. But it is difficult to see whether vertex C should be classified to community A by considering paths with length not greater than 1 (as vertex C has exactly one path with length 1 to each community). However, if we consider paths with length 2, then it is obvious that we should classify vertex C to community A.

When we choose $f_3(A) = \mathbf{I} + 0.5A + 0.25A^2$, we note from these three figures that the performance of using the correlation algorithm and that of using the mutual information algorithm are comparable. But they both are much better than the covariance algorithm. This shows that the choice of the correlation measure might also affect the performance.

B. Karate club

Now, we apply our framework to a well-known set of real-world network data, called "karate club." The set of data was observed by Wayne Zachary [28] over the course of two years in the early 1970s at an American university. During the course of the study, the club split into two groups because of a dispute

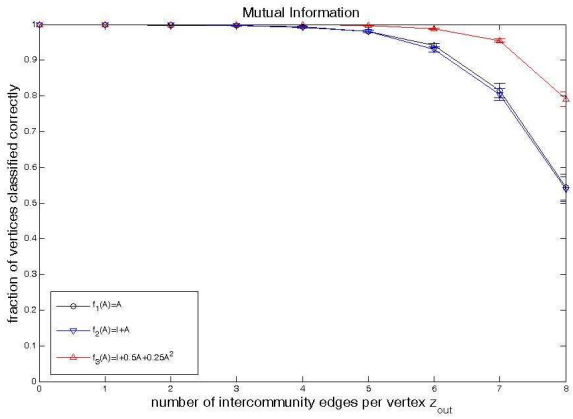

Fig. 3. Performance of $f_1(A)$, $f_2(A)$ and $f_3(A)$ under the mutual information algorithm

within the organization, and the members of one group left to establish their own club. The network of friendships between each other in the karate club observed by Zachary is shown in Figure 5.

In Figure 6, we show the dendrogram generated by using the covariance algorithm with $f_3(A) = I+0.5A+0.25A^2$. The algorithm is run until there is only one community left. As such, we can cut through the dendrogram at different levels to give divisions of the network into larger or smaller communities. As shown in Figure 6, the dendrogram generated by our algorithm matches perfectly to original structure observed by Zachary.

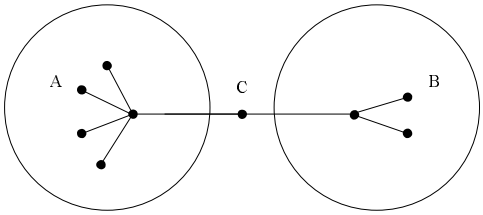

Fig. 4. An illustrating example for vertices that are difficult to classify by considering paths with length not greater than 1.

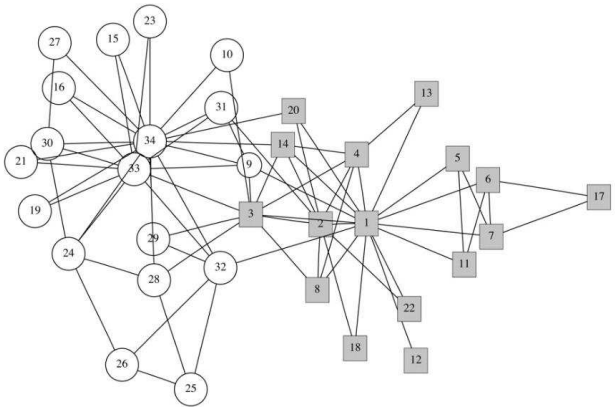

Fig. 5. The network of friendships between each other in the karate club observed by Zachary.

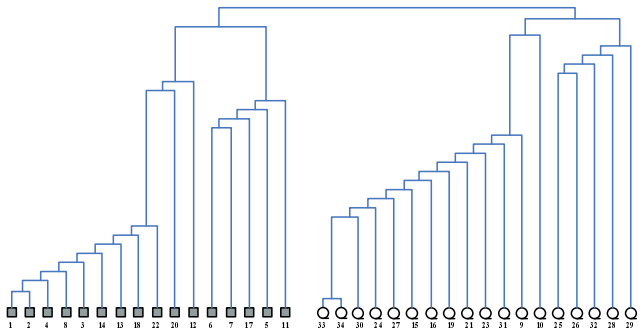

Fig. 6. The dendrogram of the karate club by using the covariance algorithm with $f_3(A) = I + 0.5A + 0.25A^2$.

This is better than the original Newman's fast algorithm, where vertex 10 is classified incorrectly. The intuition behind this is exactly the same as explained by the illustrating example in Figure 4.

VI. Conclusion

Based on Newman's fast algorithm, in this paper we developed a general probabilistic framework for detecting community structure in a network. The key idea of our framework is to characterize a graph by a bivariate distribution that specifies the probability of the two vertices appearing at both ends of a randomly path in the graph. We gave a probabilistic definition of a community and a definition of a modularity index. We also proved a couple of theoretical results for the class of distribution-based clustering algorithms. In comparison with the original Newman fast algorithm, our framework has the additional freedom to choose a bivariate distribution and a correlation measure that can be used for performance improvement. From our simulations, the choice of a bivariate distribution from a randomly selected path with length not greater than 2 performs much better than the original Newman's fast algorithm (in which an edge is selected uniformly). Finally, we note that our framework can also be easily extended to weighted networks [14] and directed networks [18]. We believe this general framework can also be used for other classes of algorithms, in particular the data compression algorithms in [23], [24].

References

[1] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20 , pp. 172-188, 2008.

[2] A. Clauset, M. E. J. Newman, C. Moore, "Finding community structure in very large networks", *Physical Review E*, 2004.

[3] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York, NY: John Wiley & Sons, 1991.

[4] I. Dhillon, , Y. Guan, and B. Kulis, "Kernel k-means, spectral clustering and normalized cuts," In *Proceedings of the 10th international conference on knowledge discovery and data mining*, pp. 551-556, 2004.

[5] L. Danon, Albert, Díaz-Guilera1, J. Duch and A. Arenas, "Comparing community structure identification," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, P09008, 2005.

[6] J. Duch and A. Arenas, "Community identification using extremal optimization," *Physical Review E* vol. 72, 027104, 2005

[7] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classficiation*, John Wiley & Sons, 2001.

[8] P. Erdös and A. Rényi, "On random graphs," *Publicationes Mathematicae Debrencen* 6, 290, 1959.

[9] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks", *Proc. Natl. Acad. Sci.* U.S.A., vol. 99, p. 7821 ,2002.

[10] Y. Hu, H. Chen, P. Zhang, M. Li, Z. Di, and Y. Fan, "Comparative definition of community and corresponding identifying algorithm," *Physical Review E*, vol. 78, 026121, 2008.

[11] Fortunato, "Community detection in graphs,"*Physics Reports*, 2010.

[12] S. Fortunato and M. Barthélemy, "Resolution limit in community dectection," *Proc. Natl. Acad. Sci.* U.S.A., vol. 104, pp. 36-41, 2007.

[13] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, vol. 69, 066133, 2004.

[14] M. E. J. Newman, "Analysis of weighted networks," *Physical Review E*, vol. 70, 056131, 2004.

[15] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks", *Physical Review E*, vol. 69, 026113, 2004.

[16] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: a kernel approach," *Machine Learning*, vol. 74, pp. 1-22, 2009.

[17] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Physical Review E*, vol. 80, 056117, 2009.

[18] E. A. Leicht and M. E. J. Newman, "Community structure in directed networks," *Physical Review Letters*, vol. 100, 118703, 2008.

[19] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory: the Mathematics of Computer Performance Modeling*. Springer-Verlag: New York, 1995.

[20] M. A. Porter, J.-P. Onnela, and P. J. Mucha "Communities in Networks," *Notices of the American Mathematical Society*, vol. 56, no. 9, pp. 1082-1097, 2009.

[21] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 101, pp. 2658-2663, 2004.

[22] U. N. Raghavan, R. Albert, S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks", *Physical Review E*, vol. 76, 036106, 2007.

[23] M. Rosvall and C. T. Bergstrom, "An information-theoretic framework for resolving community structure in complex networks," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 104, no. 18, pp. 7327-7331, 2007.

[24] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 105, no. 4 pp. 1118-1123, 2008.

[25] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 2006.

[26] F. Wu and B.A. Huberman, "Finding communities in linear time: a physics approach," *Eur. Phys. J.*, B38, pp. 331-338, 2004.

[27] R. Yuster and U. Zwick, "Fast sparse matrix multiplication," *ACM Transactions on Algorithms*, vol. 1, pp. 2-13, 2005.

[28] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, pp. 452V473, 1977.