IJST, Transactions of Electrical Engineering, Vol. 37, No. E1, pp 51-65
Printed in The Islamic Republic of Iran, 2013
© Shiraz University

# SOCIAL NETWORKS COMMUNITY DETECTION USING THE SHAPLEY VALUE[*]

A. R. HAJIBAGHERI, A. HAMZEH[**], H. ALVARI AND S. HASHEMI

School of Electrical and Computer Engineering, Shiraz University, Shiraz, I. R. of Iran
Email: ali@cse.shirazu.ac.ir

**Abstract–** As a result of the increasing popularity of social networking websites like Facebook and Twitter, analysis of the structure of these networks has received significant attention. The most important part of these analyses is towards detecting communities. The aforementioned structures are usually known with extremely high inter-connections versus few intra-connections in the graphs. In this paper, in spite of most approaches being optimization based, we have addressed the community detection problem (CDP) by a novel framework based on Information Diffusion Model and Shapley Value Concept. Here, each node of the underlying graph is attributed to a rational agent trying to maximize its Shapley Value in the form of information it receives. Nash equilibrium of the game corresponds to the community structure of the graph. Compared with the other methods, our approach demonstrates promising results on the well-known real world and synthetic graphs.

**Keywords–** Social network, Shapley value, information diffusion, Nash equilibrium, community structure

## 1. INTRODUCTION

A social network is a complex graph-based structure composed of individuals called nodes, which are connected by one or more specific types of interdependency such as friendship, common interest, financial exchange, etc. which are often called edges. Recently, online social networking websites such as Facebook, Twitter and etc., have become tremendously popular, because they let people all over the world communicate with their friends, send emails, spread opinions on some issues, etc., in the cyberspace without in person meetings. These online interactions on the Internet are provided by modern information and communication technology (ICT). As a result, the way that Social Network Analysis (SNA) is dealt with has been changed completely [1].

Graphs have often been considered as a powerful representation tool in studying social networks and their properties since the 20th century. Graph vertices and edges are respectively regarded as the paradigms of the entities in social networks (e.g. people and the interactions between them). Nowadays, the emergence of computational resources, extensive data and the recent rapid expansion of these networks to millions or even billions of vertices have produced a deep change in the way graphs are approached [2, 3, and 4]. SNA started in the 1930s and since then has become one of the most important topics in sociology [1, 5]. Social networks, like many other networks, show several interesting properties such as high network transitivity [6], power-law degree distributions [7] and the existence of repeated local motifs [8], yet the significant attribute recently under consideration is community structure' or `clustering'; the appearance of dense connected groups, modules, clusters or communities of vertices in the network graph and sparser connections between them [3]. In its simplest form, community structure refers

[*]Received by the editors July 22, 2013; Accepted May 25, 2013.

[**]Corresponding author

to the existence of modules or communities with dense inter-connections versus sparse intra-connections. A toy example network and its community structure are shown in Fig. 1.

The word community itself refers to a social context. People naturally tend to form groups within their families, work environments and friends. Communities of social networks can be friendship circles, groups of people sharing common interests and/or activities, etc. Furthermore, many other networked systems including biology and computer science, have built-in communities. This property has high applicability and therefore attracts a lot of researchers from different fields. For example, groups within the World Wide Web correspond to web pages on the related topics [9], groups in social networks like Facebook show knit relationships between their members [10] and they can be used to design reliable friend recommendation systems, or groups in a metabolic network represent cycles and other functional groupings [11]. In addition, clustering Web clients having similar interests and being geographically near each other, can improve the performance of services provided on the World Wide Web [12]. Detecting clusters of customers with similar interests in the network of purchase relationships between them and products of online retailers (e.g. Amazon 4) can lead to setting up efficient recommendation systems and improving business opportunities [13]. Moreover, clustering large graphs can help in creating data structures to store the graphs more efficiently [14].

During the last decade, a large variety of algorithms have been proposed to solve the problem of community detection. However, most of them work based on the structural attributes of the network such as number of vertices, degree of each vertex, etc. [8]. In this work we address the community detection problem as a game-theoretic approach employing Information Diffusion model and Shapley Value. As the results show, our proposed framework performs well in detecting finer community structure of the underlying graph. Our main contributions are as follows:

1. We introduce *SID* framework to analyze social network communities. In this work we address the community detection problem as a game-theoretic approach employing Information Diffusion model. To the best of our knowledge, it is the first time that this problem has been formulated in the form of Information Diffusion concept and Game theory. The results show that, the proposed framework outperforms other rival methods in detecting finer community structure of the underlying graph.
2. We extend the GADM model introduced by Lahiri and Cebrian [15] to achieve a more accurate model for Information Diffusion in social networks by defining a mutation operator. We believe sometimes, nodes can update their information without contacting other nodes in the network, similar to what happens in real life.
3. We provide an iterative algorithm that is guaranteed to converge to a promising solution. In this algorithm, each node of the graph is considered as a selfish agent that tries to maximize its total utility. The Nash equilibrium [16] of the game corresponds to the community structure of the graph.
4. In contrast to most of the existing methods, the proposed method does not rely on the structural attributes of the underlying network. We use a simple Information Matrix, which stands for the amount of information exchanged between agents. This characteristic of our method helps us to solve the problem of community detection without a need to consider basic properties of the graph such as number of nodes, degree of each node, etc.
5. There is no parameter setting in the proposed approach as opposed to the other similar methods which rely on several parameters.

The rest of the paper is organized as follows: in Section 2, a brief review of the state-of-the-art methods is presented. Then, in Section 3, we describe our proposed framework in detail. Our results on different real-world and synthetic datasets are described in Section 4. We conclude the paper in Section 5.

## 2. RELATED WORK

The problem of community detection is a long standing research appearing in various forms in several disciplines including sociology and computer science. The first analysis of community structure dates back to 1927 and the work carried out by Stuart Rice [17, 18], which looked for clusters of people in small political bodies, based on the similarity of their voting patterns. In 1955, Weiss and Jacobson searched for work groups within a government agency [19]. The authors studied the matrix of working relationships between agency members who were identified by means of private interviews. Work groups were separated by removing the members working with people of different groups who acted as their connectors. This idea of cutting the bridges between groups is at the core of several modern algorithms of community detection. Traditional techniques to find communities in social networks are hierarchical and partitioned clustering, where vertices are joined into groups according to their mutual similarity. In general, traditional techniques to find communities in social networks are hierarchical and partitional clustering, where vertices are joined into groups according to their mutual similarity.

Indeed, several works have been done in the literature which can be categorized into two main groups: optimization methods and methods with no optimization, which search for some predetermined structures. From these methods one can refer to the works done by Girvan and Newman in 2002 and 2004 introducing two important concepts ‘edge betweenness’ [10] and ‘modularity’ [20], the work of Brandes and Erlebach which coins the term ‘conductance’ [21] and the work done by Palla et al. [22]. In [10], Girvan and Newman proposed a new algorithm to identify edges lying between communities which by their successive removal, the isolation of the communities happens. The inter-community edges are detected according to the values of a centrality measure, the edge betweenness that expresses the importance of the role of the edges in processes where signals are transmitted across the graph following paths of minimal length. That work triggered a huge activity in the field where many new methods have been proposed in recent years. In particular, physicists entered the game, bringing in their tools and techniques such as spin models, optimization, percolation, random walks, synchronization and etc., which rapidly became the main ingredients of new original algorithms. The field has also taken advantages of concepts and methods from computer science, nonlinear dynamics, sociology and discrete mathematics.

In comparison with the previous discussions, very few works have been done based on the game theory (See for example [23, 14]). These works address the problem of community detection by a game-theoretic framework in which nodes of underlying social network graph are considered as rational agents who want to maximize their payoff according to some criterion. The work done in [23] can also support the overlapping concept. In this work, the difference between gain functions and loss functions is used as a utility function for each of the agents and the Nash equilibrium of the game reveals the final division of the graph. The gain function used in [23] is based on the Modularity concept [24] and the loss function is defined as a simple linear function with respect to the number of membership labels. In general, the merit of using the game-theoretic methods is that they are grounded with a systematic theory for formation of communities in the networks, as in the real world, which communities are formed based on some purposes, not for optimizing some local or global objectives.

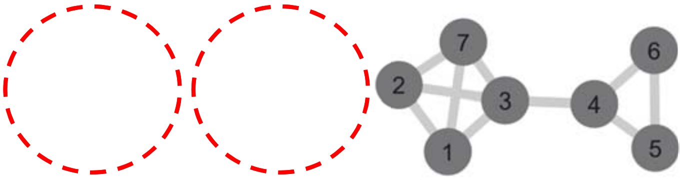


Fig. 1. Toy example network and its community structure (Each circle with dotted border shows a community)

## 3. PROPOSED METHOD

### *a) Motivation*

It is necessary for everyone to make a friendship with those who have something valuable to share. Informative human-human social interactions motivated us to assign the community formation problem in social networks to a play of interactions between their constituents, i.e. people.

Game theory is a good tool to capture both the behavior of individuals and strategic interactions among them [14], because it can model strategic interactions between rational, autonomous and intelligent agents mathematically. In this study, based on the work done by Alvari et al. [25] we consider the community detection problem as a game in a multiagent environment and ascribe each vertex of the underlying graph to an agent. These agents try to form communities based on their Shapley values by joining to communities with total useful information for them. To quantify the information exchanged between agents, we use the formulation in [15]. Finally, the Nash equilibrium of the game corresponds to the clustering of the network. In the next section, we first explain the information diffusion concept used in our framework and then our game-theoretic approach is explained in detail.

b) Information Diffusion in Social Networks

Diffusion processes take place in social networks and we can use these processes to model different phenomena in the world surrounding us such as the spread of computer and human viruses, and the information about an invention or idea. Obtaining accurate information about these processes is difficult; therefore, we will use diffusion models to describe their behavior. A diffusion model probabilistically indicates a process of diffusion takes place and spreads through a social network [15].

Information diffusion is a special kind of diffusion process in which information spreads through vertices of a social network. Here, information is considered as specific details about an innovation or an idea. There exist two well-known probabilistic models for information diffusion, the Independent Cascade model and the Linear Threshold model [18, 26, 27]. Recently, Lahiri and Cebrian [15] proposed a new model based on canonical Genetic Algorithm [28] paired with Holland's hyper-defined objective functions [29], namely GADM (Genetic Algorithm Diffusion Model). In the following, formal definition of the diffusion model is given.

Definition1.A diffusion model maps current state vector of a vertex to a new state vector. Given a graph $\boldsymbol{G = (V, E)}$ and a state vector $\boldsymbol{S_v^{(t)}}$ for every vector $\boldsymbol{v} \in \boldsymbol{V}$ at time $\boldsymbol{t}$, this model outputs a new state vector $\boldsymbol{S_v^{(t+1)}}$ for every vertex at time $\boldsymbol{t + 1}$ based on the state of all interacting vertices [15].

We now explain the GA Diffusion Model (GADM) [15] which we will use in our proposed community detection algorithm. Lahiri and Cebrian showed that canonical Genetic Algorithm, which uses binary string chromosomes and one-point crossover, can be used as a model for information diffusion process. Their work deals with dynamic social networks where a set of vertices $\boldsymbol{V = \{v_1, \dots, v_n\}}$ interacts over $\boldsymbol{T}$ time periods. Furthermore, a mapping exists between these individuals and chromosomes in the GA population. Chromosomes are state vectors of individuals at each time step $\boldsymbol{t}$. These state vectors are binary string with length $\boldsymbol{\beta}$. Initial state vectors can be set to zero or one can use a random distribution to initialize them. Additionally, an objective function $\boldsymbol{f(x)}$ which assigns a score to each vertex state vector is needed. For each edge $\boldsymbol{(u, v)}$ in the social network at time step $\boldsymbol{t}$, logic of canonical genetic algorithm is applied to the corresponding chromosomes in the GA population. GADM algorithm is shown in Algorithm 1. In this algorithm, state vectors are modified and adopted based on crossover operator and objective function, resulting in the occurrence of diffusion process in the social network. Obviously, we have a missing component here, and that would be an objective function to bring meaning to the mapping between the state vectors of a node and chromosomes in GA population.

Algorithm 1. **GADM**

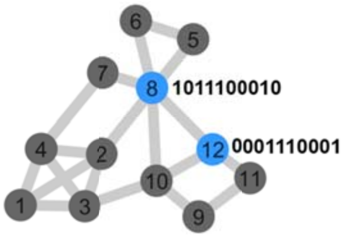
```
Input: Initialize state vectors of nodes $u$ and $v$ to $S_v$and $S_u$.
Output: New state vector for nodes $u$and $v$.
Repeat{
Set $S_v^{(t+1)} = S_v^{(t)}$and $S_u^{(t+1)} = S_u^{(t)}$.
Select a random crossover point c between $[1, \beta]$.
Create $y_1$and $y_2$by swapping the tails of $S_v^{(t)}$and $S_u^{(t)}$ where the tail is defined as all positions including and after index c.
Update state vectors: $S_v^{(t+1)} = argmax_{x=\{S_v^t, y_1, y_2\}} f(x)$and $S_u^{(t+1)} = argmax_{x=\{S_u^t, y_1, y_2\}} f(x)$.
Until (All interactions are checked)
```

To fill the aforementioned gap, Holland hyper-defined objective functions (HDFs) [29] which are synthetic objective functions can be used in GADM algorithm. HDFs are constructed from a predefined schema, a short substring with wildcards starting at a specific position. Schemas take place at random positions within the strings. At first, schemas are relatively short and we call them order 1 schemas. Pairs of such schema are concatenated to generate order 2 schema and so on. Moreover, each schema receives an individual positive or negative score generated randomly from some range. At this point, HDFs take binary strings as input and return an objective score that is the sum of the scores of all the individual schemas it contains[15].

The following is an example demonstrating the generation of a simple HDF which takes a binary string of length $n = 10$ as an input. The asterisk (‘*’) shows a character that matches both zero and one (‘don’t care’ character).

Suppose that we have a toy network example shown in Fig. 2. Now we can evaluate each node’s state vector using the HDF discussed above. As an instance, evaluation of nodes 8 and 12 objective values is depicted in the figure:

*********01* score 1, order 1 schema**
*****11***** score 1, order 1 schema**
**10******** score 2, order 1 schema**
**10*11***** score -2, order 2 schema**
**10*****01* score 3, order 2 schema**



**1011100010**
**score: 1+1+2-2+3=5**
**0001110001**
**score: 1**

**(a)** **(b)**

Fig. 2. (a) A toy network example with 13 nodes. (b) Evaluation of nodes 8 and 12 objective values

In GADM, each schema is a ‘unit’ of information and different schema carries different values. As we mentioned before, every individual in a social network has a state vector. These state vectors can contain different schema’s, which means every individual carries certain pieces of information. Each time a pair of nodes interacts with each other, they randomly exchange information based on the crossover operator. This process could result in either both nodes gaining more information than they had before or no benefit from interaction.

There is a key difference between GADM and earlier studies on information diffusion such as [18] [26] [27]. GADM is able to model the propagation and spread of multiple 'units' of information from individual to individual. These different units can interact in non-linear and complex ways based on a randomly generated HDF in order to affect the total 'information value' of each individual. The objective is to estimate the average information value of each node after multiple random HDFs and state vector initializations, in order to determine whether every individual in our social network is positioned to receive the same amount of information as a result of their interactions with other individuals.

Lahiri and Cebrian [15] used GADM to model information flow between people as they exchange emails. To test their method they used Enron email dataset. Their results indicated that only a small portion of vertices in a social network receive more information than others, regardless of how much they start with. This term is called 'information elitism' and they showed that it is not simply related to a trivial network property such as degree. To put it simply, one might assume that a vertex with high income degree may contain more information than others, but the correlation between the final information value of nodes and their degree shows absolutely the opposite. As a result, a community detection method based on this feature, is independent from network structure.

### *c) Framework*

In this section, we provide our proposed information diffusion model (Extended GADM or EGADM), how we use information diffusion in our model and rigorous mathematical formulation of our framework. We need to introduce some notations and parameters used in our work as shown in Table 1.

Table 1. Definitions of symbols

| **Sym.** | **Definition** |
|---|---|
| $G$ | Undirected and unweighted graph |
| $n$, $m$ | Number of nodes and edges |
| $S$ | Set of strategies of all agents |
| $s_i$ | Strategy of agent *i* |
| $\Phi_i$ | Shapley value of agent *i* |
| $\mathbf{I}$ | Information Matrix |
| $I_{ij}$ | Information that agent *i* received from agent *j* |
| $\delta_{ij}$ | Equals 1 if agents *i* and *j* are in the same community |

In order to use 'information elitism' phenomena in our method, we have to store information that is exchanged between nodes. Each node can either receive information from others or give them new units of information through crossover operation. Here, the interpretations of $\boldsymbol{S_v^{(t)}}$and $\boldsymbol{S_v^{(t+1)}}$ are different from what is actually used in GADM. $\boldsymbol{S_v^{(t)}}$stands for the state vector of node $\boldsymbol{v}$ before an interaction with node $\boldsymbol{u}$ while $\boldsymbol{S_v^{(t+1)}}$ is the state vector of node $\boldsymbol{v}$ after the interaction with node $\boldsymbol{u}$. In addition to the crossover operator used in GADM, we use mutation operator in order to make the information diffusion model stronger. We believe that, sometimes, nodes can update their information without contacting other nodes in the network and that is what happens in real life. The new information diffusion model algorithm is shown in Algorithm 2.

To avoid the bias of randomness we run multiple trials with different HDFs and state vectors initialization. In each run, we store the information exchanged between two nodes in a matrix called Information Matrix I. These values are normalized with respect to the maximum exchanged value after each run. What we need over multiple trials is the average of these values which we call 'average normalized information' or ANI:

$$\boldsymbol{I} = \frac{\boldsymbol{I}^{(1)}+\cdots+\boldsymbol{I}^{(n)}}{n} = \begin{bmatrix} \frac{I_{11}^{(1)}+\cdots+I_{11}^{(n)}}{n} & \cdots & \frac{I_{1n}^{(1)}+\cdots+I_{1n}^{(n)}}{n} \\ \vdots & \ddots & \vdots \\ \frac{I_{n1}^{(1)}+\cdots+I_{n1}^{(n)}}{n} & \cdots & \frac{I_{nn}^{(1)}+\cdots+I_{nn}^{(n)}}{n} \end{bmatrix} \tag{1}$$

where superscripts show the information matrix of i$^{\text{th}}$ iteration and I is the final Information Matrix.

Algorithm 2. **EGADM**

```
Input: Initialize state vectors of nodes $u$ and $v$ to $S_v$and $S_u$.
Output: New state vector for nodes $u$and $v$.
Repeat{
    Set $S_v^{(t+1)} = S_v^{(t)}$and $S_u^{(t+1)} = S_u^{(t)}$.
    Select a random crossover point c between $[1, \beta]$.
    Create $y_1$and $y_2$by swapping the tails of $S_v^{(t)}$and $S_u^{(t)}$ where the tail is defined as all positions including and after index c.
    Update state vectors: $S_v^{(t+1)} = argmax_{x=\{S_v^t,y_1,y_2\}} f(x)$and $S_u^{(t+1)} = argmax_{x=\{S_u^t,y_1,y_2\}} f(x)$.
    Do mutation on each bit of $S_v^{(t+1)}$ and $S_u^{(t+1)}$ based on mutation probability $P_m$.
}
Until (All interactions are checked)
```

Regarding what has been discussed so far, we put each vertex down to a rational agent that just thinks about maximizing its Shapley value. In doing so, whenever it is selected from a pool of agents, it periodically makes personal decisions while it is involved in the game. Later on this agent decides whether to join a new community, leave one of its communities or switch from a community to a new one.

After all, new Shapley value for the selected agent is calculated according to (2) and its old value is replaced by the new one.

$$v_i(S) = \frac{1}{m}\sum_{j=1, j\neq i}^{n} I_{ij}\delta_{ij} \tag{2}$$

where $v\colon 2^C \to \mathbb{R}$ is a characteristic function that assigns a value to each subset of *C*. Here *C* is a community which consists of one or more nodes. For this agent the Shapley value is defined as:

$$\Phi_i(v) = \frac{1}{n!}\sum_{\pi\epsilon\Omega}\left[v(P_i(\pi) \cup i) - v\big(P_i(\pi)\big)\right] \tag{3}$$

where $\Omega$ is the power set over *C* and $P_i(\pi)$ is the set of players appearing before the $i_{\text{th}}$ player in set $\pi$. In this framework, the best response strategy of an agent *i* with respect to the strategies $S_{-i}$ of other agents is calculated by:

$$argmax_{s_i' \subseteq [k]} \Phi_i(S_{-i}, s_i') \tag{4}$$

The strategy profile *S* forms a pure Nash equilibrium of the community formation game if all agents play their best strategies. In other words, in Nash equilibrium no agent can improve its own utility by changing its strategy; that is each agent is satisfied with the current utility:

$$\forall i, s_i' \neq s_i, \Phi_i(S_{-i}, s_i') \leq \Phi_i(S_{-i}, s_i) \tag{5}$$

Since reaching global Nash equilibrium is not feasible in this game, we used local Nash equilibrium [30]. The strategy profile *S* forms a local equilibrium if all agents play their local optimal strategies. Here *ls*($s_i$) refers to local strategy space of agent *i*:

$$\forall i, s_i' \in ls(s_i), \Phi_i(S_{-i}, s_i') \leq \Phi_i(S_{-i}, s_i) \tag{6}$$

Finally, the algorithm of the proposed method, namely SID, is shown in Algorithm 3.

```
Algorithm 3.SID
Input: underlying network graph G.
Output: community as a final division of G.
Initialize each node of G as an agent.
Initialize community as a set of all communities.
I = EGADM(G) or GADM (G)  //Create Information Matrix
Repeat{
    Choose a random agent from pool of agents.
    Choose the best operation among join, leave, switch or no operation according to (4).
}
Until (local Nash equilibria is reached)
```

## 4. EXPERIMENTS

We have mentioned earlier that the proposed approach can perform well in detecting communities in social networks. To illustrate this, we now present our experimental results on the well-studied real-world datasets and two popular synthetic networks. We compare our algorithm with three decentralized algorithms and one centralized algorithms. We use two evaluation metrics to show the performance of our algorithm: Normalized Mutual Information (NMI) [31] and Modularity $Q$ [32]. Table II shows algorithms and parameter settings we have used for our experiments respectively. We have implemented all of the algorithms in JAVA on a system with 4G of RAM and Intel CPU 2.53 GHz for the purpose of fair comparison.

Table 2. Algorithms used for experiments and their parameters (D=Decentralized, C=Centralized)

| Algorithm | Type | Parameters |
|---|---|---|
| HA [33] | D | $m = 0.1, \delta = 0.05$ |
| MMC [34] | D | $\alpha = 2.65, \beta = 2, \rho = 0.9, \mu = 1.08, \eta = 0.7, \gamma = 0.1$ |
| InfoMap [10] | C | None |
| LPA [35] | D | None |

LPA and HA are different varieties of basic label propagation methods while MMC is a label propagation method which uses labels to show information. InfoMap is a centralized method which is based on “unfolding” technique.

### *d) Evaluation*

*1. Normalized mutual information (NMI) [31]:* We often use normalized mutual information to measure the similarity between found partition and the ground trust partition. Assume that we have two partitions called P and Q. We form a matrix N in$|\mathbf{P}| \times |\mathbf{Q}|$.

Element $N_{ij}$ of the matrix **N** shows the number of common nodes of the i$^{th}$ community of $P$ and j$^{th}$ community of $Q$. In addition, we show the sum of i$^{th}$ row with $N_{i.}$ and the sum of the j$^{th}$ column with $N_{.j}$. NMI is in range [0, 1] and is defined as follows:

$$I(P,Q) = \frac{-2\sum_{i=1}^{|P|}\sum_{j=1}^{|Q|} N_{il}\log(\frac{N_{ij}N}{N_{i.}N_{.j}})}{\sum_{i=1}^{|P|} N_{i.}\log\left(\frac{N_{i.}}{N}\right)+\sum_{j=1}^{|Q|} N_{.j}\log(\frac{N_{.j}}{N})} \tag{7}$$

*2. Modularity [32]:* As stated above, we can use NMI only when the ground trust structure is provided. However, in some special cases, we have to test the algorithm on some synthetic datasets with no ground trust and that's when modularity will come in handy.

Although this measure has drawbacks and becomes unreliable when our networks are too sparse [25], modularity is the most popular qualitative measure in detecting communities in social networks. Formally modularity is defined as follows:

$$Q = \sum_{s=1}^{k}[\frac{l_s}{L} - (\frac{d_s}{2L})^2] \tag{8}$$

where $k$ is the number of revealed communities, $L$ denotes the total number of edges, and $d_s$ is the sum of degrees of nodes in community $S$.

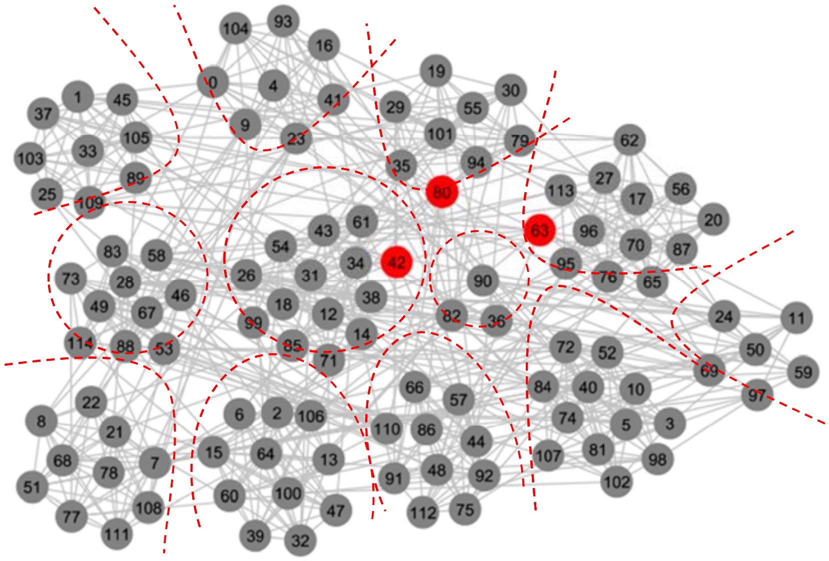


Fig. 3. SID best result on American College Football network

### *e) Dataset*

We explain our results on three well-known real-world benchmark networks and two synthetic networks in this section. Experimental results on these datasets demonstrate that our approach performs well, both on real-world and synthetic datasets.

### 1. Benchmark Networks:

*a) American College Football Network:* As a final test on real-world data, we turn to the American college football network which represents the schedule of games during the 2000 season. This network was previously used by Girvan and Newman (M. Girvan and M.E.J. Newman 2002) and its community structure is known. The network contains 115 nodes and 616 edges. Each node represents a football team and each edge shows a game between two teams connected to each other. Teams are divided into 12 conferences and we can consider each conference as a community in this case. It occurs that games held between teams of the same conferences are more than games played between different conferences.

The results on this network indicate that the proposed approach detects most conferences correctly except for a few teams, Connecticut (42) and Navy (80) from IA, Independent and Middle Tennessee State (63) from Sunbelt. To the best of our knowledge, the reason for this misclassification is that these teams play more games with teams in other conferences rather than with the teams in their own groups and in this way they receive more information from other associations. As a result, these teams tend to join associations with which they play more games. To overcome this problem in our algorithm and as a future work, one can use a function to control each node decision for joining new communities. Fig. 3 shows communities found by our algorithm on the American college football network. The performances of a variety of algorithms on this network in the form Modularity and NMI are depicted in Table 3.As it can be understood, SID performs better than other methods on this dataset. Results shown in Table 3 are average from 100 runs.

Table 3. The Average Results of 100 Runs By SID, HA, MMC, LPA and infomap

| | **Q** | | | **NMI** | | |
|---|---|---|---|---|---|---|
| | ***Dolphin*** | ***Football*** | ***Karate*** | ***Dolphin*** | ***Football*** | ***Karate*** |
| SID | **0.538** | **0.598** | 0.373 | **0.715** | 0.838 | **1.00** |
| HA | 0.449 | 0.595 | 0.300 | 0.707 | 0.885 | 0.754 |
| MMC | 0.526 | 0.597 | 0.371 | 0.579 | **0.927** | 1.00 |
| AOC-d | 0.447 | 0.575 | 0.368 | 0.555 | 0.899 | 0.761 |
| LPA | 0.450 | 0.591 | 0.362 | 0.710 | 0.892 | 0.751 |

*b) Dolphin Network:* The dolphin network represents the relationships of 62 bottlenose dolphins, introduced by Lusseau [36] consisting of 62 nodes and 159 edges. Based on Lusseau's observations, these dolphins were divided into two groups because of some reasons. The performances of a variety of algorithms on this network in the form Modularity and NMI is depicted in Table III.Fig. 4 shows communities found by our algorithm on the Dolphin network.

*c) Zachary Karate Club Network:* The Zachary karate club network [37] which consists of 34 nodes and 78 edges shows relations between these nodes standing for club members. This network is widely used as a benchmark for community detection algorithms. The Zachary karate club network is divided into two roughly equal-sized groups due to the disagreement between club administrator and the principal karate teaher, represented by node 34 and node 1 respectively. Our approach discloses these two groups perfectly. Figure 5 shows communities found by our algorithm on the Zachary network.

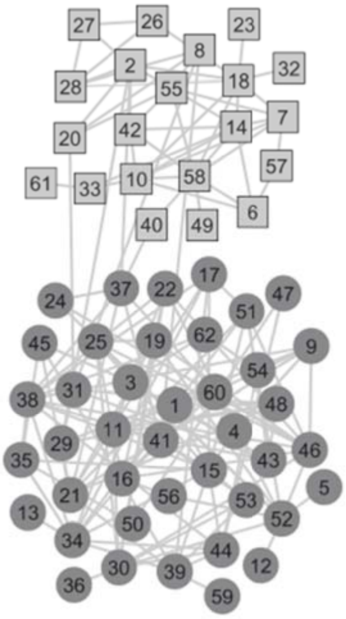


Fig. 4. GID best result on dolphin dataset (Divided into two groups correctly with NMI=1.00)

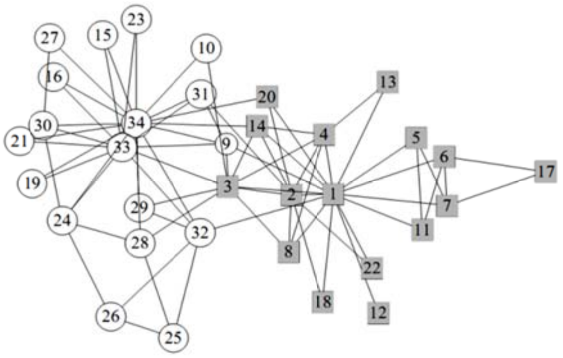


Fig. 5. Zachary karate club ground truth. reprinted from ref (Newman and M. Girvan 2004)

### *2. Synthetic networks:*

*a) GN Synthetic Networks:* GN synthetic network is probably the most popular community generation model [38]. This model contains four communities and 128 nodes in which each node has the same expected degree-16. There exists a mixing parameter **μ** which controls the ratio between the external degree with respect to community and degree of a node. The results shown in Fig. 2 are the performances of SID, MMC, LPA,HA and InfoMap. Every point in this figure is the average of 100 runs for each algorithm.

The number of communities is fixed to four in GN sythetic networks and our algorithm detects only four communities in each run while other algorithms such as Sharc sometimes finds more than four communities. As we can see in Fig. 6, the proposed approach performs extremely well when the value of mixing parameter is lower than 0.45. Finally, the NMI value approaches to zero when we set the mixing parameter to 0.5, similar to HA and LPA.

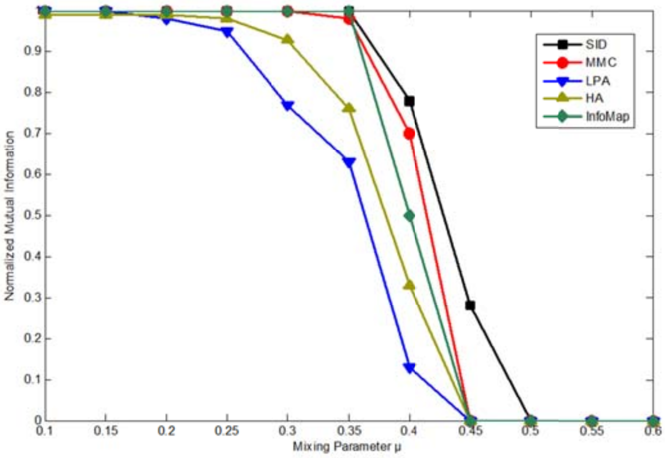
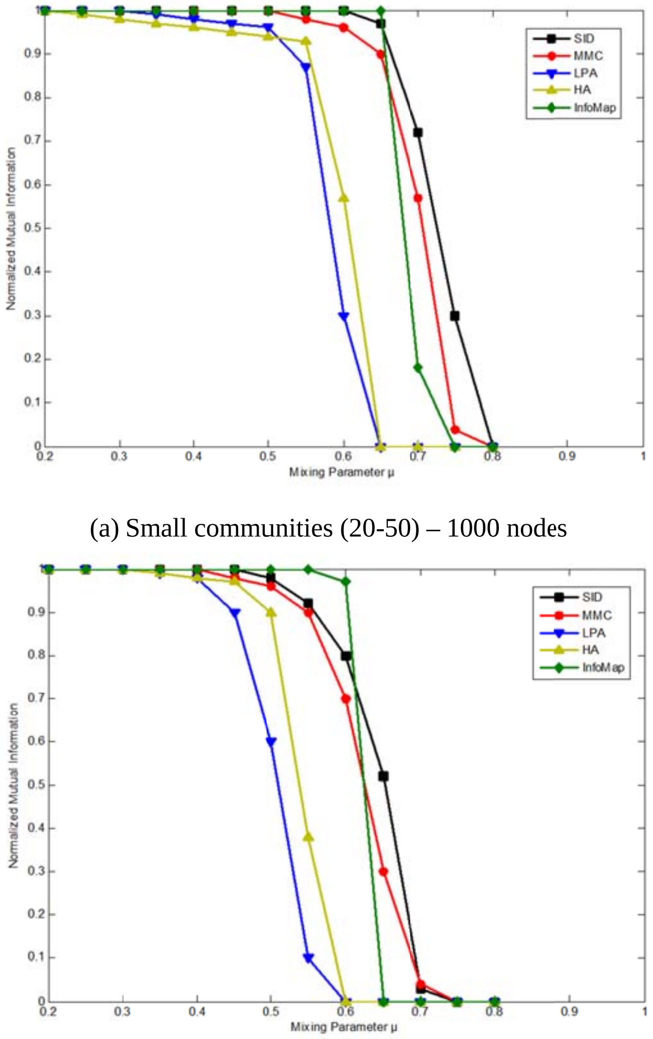


Fig. 6. Algorithms Performances on GN Synthetic Network

*b) LFR Synthetic Networks:* Recently Lancichinetti [39] proposed a method to generate synthetic networks in which we can control both, the distribution of degrees and communities sizes. This attribute of LFR method covers the GN synthetic networks drawbacks and, as a result, it is becoming a commonly used method to generate synthetic networks. It is realistic, reasonable and also able to generate networks with overlapping communities. For this experiment, the maximum degree of each node is 50 and the average degree is 20. The exponent of the degree distributon is set to -2. Communities have between 20 to 50 nodes and 20 to 100 nodes and we call them small communities and big communities respectively.The performances of different algorithms compared with our algorithm, applied on two different networks with small and big communities is shown in Fig. 7.As can be seen in this figure, SID performs better than almost all the other methods when $\boldsymbol{\mu \leq 0.7}$, while LPA and HA reach zero at $\boldsymbol{\mu = 0.65}$. When $\boldsymbol{\mu > 0.75}$, nearly all of algorithms perform equally.



(a) Small communities (20-50) – 1000 nodes

(b) Big communities (20-100) – 1000 nodes

Fig. 7. Algorithms Performances on LFR Synthetic Network

***c***) ***Erdös − Réyni***_Networks:_ Finally, we conduct an experiment on Erdös-Réyni random networks. This method generates random networks which contain no communities and meaningful relationships between the nodes. Lancichinetti et al. (Lancichinetti and Fortunato 2009) mentioned that it is essential for a community detection algorithm to identify a random network with no community structure. In this experiment, the network takes different sizes of 100, 500 and 1000. The results show that GID considers all nodes as a single community while other algorithms except MMC group nodes into several small size communities. Although MMC gets the same result as we do, its results depend strictly on several input parameters.

### *f) Time Complexity*

SID is able to handle real-world social media datasets in a reasonable time period. As it can be seen in Table 4, the computational time of SID running on Flickr and BlogCatalog datasets is about 7 hours and 2 hours respectively. Despite the high computational cost of SID, our method deals well with changes in datasets structure. In this case, it is not necessary to run SID on the whole network from scratch and we only need to compute information exchanges between newly added nodes and the ones that communicate with them. Since community detection is usually a time-consuming task in large social media datasets and it becomes worse if an algorithm is not able to deal with relatively small changes. The aforementioned feature of SID helps us to overcome this drawback.

The total order of our framework is $\boldsymbol{O}\ (\boldsymbol{K} \times \boldsymbol{m} + \boldsymbol{C} \times \boldsymbol{n} \times \boldsymbol{D})$. SID consists of two phases: EGADM (Offline) and Game Theoretic Framework (Online). The first part needs $\boldsymbol{O}\ (\boldsymbol{K} \times \boldsymbol{m})$ where $\boldsymbol{K}$ is the number of multiple runs of EGADM to avoid randomness is. On the other hand, the second part of the algorithm needs $\boldsymbol{O}\ (\boldsymbol{C} \times \boldsymbol{n} \times \boldsymbol{D})$ where $\boldsymbol{D}$ is the degree of each node and $\boldsymbol{C}$ is the number of times each node is selected for personal decision.

### *g) Mega networks*

Also, as an additional experiment, to show applicability of our method in real-world networks, we apply our method on a random sample of Facebook database with 1,000,000 nodes. It must be mentioned that community detection needs to be validated using known and existing communities and in this sample database, we could not access such data to evaluate the NMI. However, this part of our experiment is interesting for us only for the scalability point of view. So, the accuracy is not the matter here. Our method can detect 22371 communities in only 35 seconds using an Intel cori7 processor and 16GB of memory. It is somehow promising to show that the method is completely scalable and can be applied on Mega networks easily.

## 5. CONCLUSION AND FUTURE WORK

In this study, we have proposed a Game theoretic framework based on Shapley value of the vertices and Information Diffusion Model to identify community structure of the underlying network graph. In this framework each node of the graph is considered as an agent that calculates Shapley value for each community it has a connection with. Specifically, each selected agent chooses between *join*, *leave* and *switch* operations iteratively. The Nash equilibrium of the game corresponds to the community structure of the network.

The results demonstrate our method's superiority over other well-known methods. For future work, the proposed framework can be easily extended to be used in dynamic social networks. In addition, different models for information diffusion in social networks can be used.

## REFERENCES

1. Wasserman, S. (1994). *Social network analysis: Methods and applications*. Cambridge University Press, Cambridge.
2. Albert, R. & Barabási, A. (2002). Statistical mechanics of complex networks. *Reviews of modern physics*, Vol. 74, No. 1, p. 47.
3. Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM review*, Vol. 45, No. 2, p. 167256.
4. Barrat, A., Barthelemy, M. & Vespignani, A. (2008). *Dynamical processes on complex networks*. Cambridge University Press, New York, NY, USA.
5. Scott, J. (2000). *Social network analysis: A handbook*, Sage.
6. Watts, D. J. & Strogatz, S. H. (1998). Collective dynamics of small-world networks. *Nature*, Vol. 393, No. 6684, p. 440442.
7. Barabási, A. L. & Albert, R. (1999). Emergence of scaling in random networks, *Science*, Vol. 286, p. 509512.

8. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. & Alon, U. (2002). Network motifs: Simple building blocks of complex networks, Science 298: 824827.
9. Flake, G. W., Lawrence, S. R., Giles, C. L. & Coetzee, F. M. (2002). *IEEE Computer*, Vol. 35, pp. 66-71.
10. Girvan, M. & Newman, M.E.J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, Vol. 99, No. 12, pp. 7821–7826.
11. Chen, J. & Yuan, B. (2006). Detecting functional modules in the yeast protein interaction network. *Bioinformatics*, Vol. 22, 22832290.
12. Krishnamurthy, B. & Wang, J. (2000). On network-aware clustering of web clients. SIGCOMM Computer Communication. Rev.30: 97110.
13. Reddy, K. P., Kitsuregawa, M., Sreekanth, P. & Rao, S. S. (2002). *In DNIS '02: Proceedings of the Second International Workshop on Databases in Networked Information Systems(Springer-Verlag, London, UK),* pp. 188-200.
14. Wu, A Y., Garland, M. & Han, J. (2004). Mining scale-free networks using geodesic clustering. KDD04, pp. 719724.
15. Lahiri, M. & Cebrian, M. (2010). The genetic algorithm as a general diffusion model for social networks. *Proc. of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010)*, Atlanta, Georgia.
16. Eftekhari, M. & Eghbali, H. J. (2006). Strategic bidding with regard to demand elasticity. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 30, No. B6, pp. 691-700.
17. Rice, S. A. (1927). The identication of blocs in small political bodies. *The American Political Science Review*, Vol. 21, 619627.
18. Alós-Ferrer, C. & Ania, A. (2001). Local equilibria in economic games. *Econ Lett*, Vol. 70, No. 2, pp. 165-173.
19. Weiss, R. S. & Jacobson, E. (1955). A method for the analysis of the structure of complex organizations. *American Sociological Review*, Vol. 20, No. 6, 661668.
20. Newman, M. E. J. & Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, Vol. 69, No. 2, 026113.
21. Brandes, U. & Erlebach, T. (2005). Network Analysis, Springer-Verlag Berlin/Heidelberg, [New York].
22. Palla, G., Derenyi, I., Farkas, I. and Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society, Nature 435(7043): 814818.
23. Chen, W., Liu, Z., Sun, X. and Wang, Y. (2010). A game-theoretic framework to identify overlapping communities in social networks.
24. Newman, M E J. (2006). Modularity and community structure in networks, Proceedings of the National Academy of Sciences 103(23): 85778582.
25. Alvari, H., Hashemi, S. & Hamzeh, A. (2011). *Detecting overlapping communities in social networks by game theory and equivalence concept*. AICI 2011, Part II, LNAI 7003, pp.620 - 630, Springer-Verlag
26. Gruhl, D., Guha, R., Liben-Nowell, D. & Tomkins, A. (2004). Information diffusion through blogspace. *SIGKDD Explorations*, Vol. 6, pp. 491–501.
27. Fushimi, T., Kawazoe, T., Saito, K., Kimura, M. & Motoda, H. (2008). What does an information diffusion model tell about social network structure. Pacific Rim Knowledge Acquisition Workshop, pp. 122–136.
28. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
29. Holland, J. H. (2000). Building blocks, cohort genetic algorithms, and hyperplane-defined functions. *Evol. comp*., Vol. 8, No. 4, pp. 373–391.
30. Alós-Ferrer, C. & Ania, A. (2001). Local equilibria in economic games. *Econ Lett*, Vol. 70, No. 2, pp. 165-173.
31. Danon, L., Diaz-Guilera, A., Duch, J. & Arenas, A. (2005). Comparing community structure identification. *Journal of Statistical Mechanics:Theory and Experiment*, Vol. 9, p. 8.

32. Newman, M. E. J. & Girvan, M.  (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, Vol. 69, No. 2, p. 026113.
33. Leung, I. X. Y., Hui, P., Lio, P. & Crowcroft, J. (2008). Towards real-time community detection in large networks. *Phys. Rev. E*, Vol. No. 6 Pt 2, p. 10.
34. Chen, M. (2011). Discovering communities by information diffusion. (2011). *Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD).*
35. Raghavan, U. N., Albert, R. & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E*, Vol. 76, No. 3, p. 036106.
36. Lusseau, D.  (2003). The emergent properties of a dolphin social network. *Proceedings of the National Academy of Sciences*, Vol. 270 Suppl, No.0962-8452, pp. S186–8.
37. Zachary, W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, Vol. 33, pp. 452–473.
38. Lancichinetti, A. & Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Phys. Rev. E*, Vol. 80, No. 5, p. 056117.
39. Lancichinetti, A. & Fortunato, S. (2009). Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys Rev E*, Vol. 80, No. 1, 16118.