Discovering overlapping communities in social networks: A novel game-theoretic approach

Hamidreza Alvari *, Sattar Hashemi and Ali Hamzeh
Department of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran
E-mails: alvari@cse.shirazu.ac.ir, s_hashemi@shirazu.ac.ir, ali@cse.shirazu.ac.ir

Abstract. Identifying communities in social networks has been receiving the increasing attentions recently. However, the overlapping concept has received little attentions in the literature, although it is observed in almost all social networks. In this study, we propose a framework based on the game theory and the structural equivalence concept to address the detection of overlapping communities in social networks. Specifically, we consider the underlying graph as a hypothetical social networking website and regard each vertex of this graph as an agent performing in this multiagent environment. Since each agent may belong to several communities simultaneously, we are able to find overlapping community structure of social networks. The rigorous proof of the existence of Nash equilibrium in this game is provided which shows that the method always reaches to the final solution. Experimental results on the benchmark and real world graphs show superiority of our approach over the other state-of-the-art methods.

Keywords: Game theory, multiagent environment, overlapping communities, social networks, structural equivalence

1. Introduction

A social network is a complex network demonstrating social interactions between people [36]. Recently, online social networking websites such as Facebook, Twitter, Myspace, have become tremendously popular, because they let people all around 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 potent representation tool, graph and its literature has become a renowned component in studying social networks and their properties since 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 recent rapid expansion of these networks to million or even billion of vertices have produced a deep change in the way graphs are approached [2,5,25].

Social network analysis (SNA) has been started in 1930s and since then has become one of the most important topics in sociology [34,36]. Social networks like many other networks, show several interesting properties such as high network transitivity [37], power-law degree distributions [4] and the existence of repeated local motifs [24], 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 [25].

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 World Wide Web correspond to web pages on the related topics [11], groups in social

*Corresponding author. E-mail: alvari@cse.shirazu.ac.ir.
networks like Facebook, show knit relationships between their members [14] and they can be used to design reliable friend recommendation systems or groups in a metabolic network represent cycles and other functional groupings [8]. 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 [18]. Detecting clusters of customers with similar interests in the network of purchase relationships between them and products of online retailers (e.g. Amazon⁴) can lead to set up efficient recommendation systems and improve business opportunities [32]. Moreover clustering large graphs can help in creating data structures to store the graphs more efficiently [39].

Given an underlying graph of network, the community detection problem is usually defined as clustering these vertices into communities or groups based on some predefined measures where these communities can have intersections (i.e. overlapping communities) or not. However, this problem, intuitive at first sight, is not well defined because its main components (e.g. the community concept) are not precisely defined and there are often some ambiguities in their definitions. This has resulted in presenting so many interpretations in the literature (see [12] for an extensive analysis).

The main challenging issue on community detection problem is indeed the lack of quantitative measures and a formal definition of community. Unfortunately there is no universally accepted definition in hand because such a definition often depends on the underlying problem and its application. But intuitively, we expect more edges inside the community compared to the outside of it and the rest of the graph. In other words, intra-connection edges must always be more than inter-connection edges. This simple concept is the core of nearly all community definitions.

Social network analysts have distinguished three main classes of definitions for community: local, global and vertex similarity based [12]. According to the local definition, a community is being evaluated independently from the graph, while in the global definition communities are defined with respect to the graph as a whole. Definitions based on the vertex similarity, assume that communities are groups of vertices most similar to each other.

As a current research, one of the new trends in social networks analysis is towards the overlapping concept. This means that some of nodes may belong to more than one community simultaneously. This is true even in our social lives; we often belong to our families, friends, colleagues, etc., at the same time. Nevertheless, this challenging issue is addressed in a few recent works in the literature and most methods can only detect standard partitions, i.e. partitions in which each node is assigned to a single community.

In this work, motivated from human incentives to interact with each other and establish friendships, we consider the problem of finding overlapping communities in social networks as a game-theoretic approach in a hypothetical social networking website and ascribe each vertex of the underlying social network graph to a user performing in this multiagent environment. In this environment, each agent tries to maximize its total utility by establishing friendships with its similar agents through two general steps: First, it sends friend requests to its similar agents and second it receives friend suggestions from the system, where the similarity between each pair of agents is calculated by Pearson correlation as a structural equivalence concept. The more similar the agents are, the higher the utility is. The final community structure of the underlying social network graph is being revealed when the game reaches the Nash equilibrium. Once all of the agents are satisfied with their utilities and therefore no agent wants to change its strategy, the final communities are formed. Experiments show that this framework is able to find finer community structure of the underlying graph, compared to the other similar methods in the literature. Our contributions are then twofold:

- We enrich the model introduced in [9] to imitate the behavior of social networking websites environments and provide a better view of social dynamics. The enriched model is able to detect finer and more accurate community structures at the expense of performing more operations and running time. Specifically, we provide an iterative algorithm consisting of two phases: Personal and System phases, where each node of the graph is regarded as a selfish agent who tries to maximize its total utility. We show that this method is guaranteed to converge to a promising solution and does not need to set specific parameters. The Nash equilibrium of the game corresponds to the community structure of the underlying graph.
- Compared to the existing model in [9], two major differences can be immediately figured out. (1) Instead of employing Modularity based function [9] to calculate utilities for agents, we embed Pearson similarity function in our work which

⁴[www.amazon.com](http://www.amazon.com).
has a sensible interpretation borrowed from real life. This allows us to consider similarity quantities in calculating agents utilities. (2) We propose two complex communal decision based operators suggestion and eviction both taking root in real life which help our model converges to a model best representing the cyberspace environment of existing social networking websites.

What comes hereinafter is organized as follows. Section 2 describes some of the state-of-the-art methods in the literature. In Section 3 we delve into the problem by introducing our proposed framework. The experimental results are explained in Section 4 and finally we conclude the paper in Section 5. In the Appendix, we provide mathematical statements to guarantee the existence of the Nash equilibrium in the game of concern.

2. Related works

Community detection problem (CDP) has a long history and has appeared in various forms in several disciplines including sociology and computer science. Perhaps, the first analysis of community structure dates back to 1927, where Stuart Rice looked for clusters of people in small political bodies, based on the similarity of their voting patterns [33]. Two decades later, George Homans showed that social groups could be revealed by suitably rearranging the rows and the columns of matrices describing social ties, until they take an approximate block-diagonal form [17]. In 1955, Weiss and Jacobson searched for work groups within a government agency [38]. 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.

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, many works have been done in the literature, which can be categorized into two main groups: optimization methods which look for optimizing some measures and methods with no optimization, which search for some predetermined structures. From these works, one can refer to the works done by Girvan and Newman in 2002 and 2004 introducing two important concepts “edge betweenness” [14] and “modularity” [28], the work done by Brandes and Erlebach coining the term “conductance” [7] and the work done by Palla et al. [31]. In [14], Girvan and Newman proposed a new algorithm, aiming at the identification of edges lying between communities and their successive removal that after some iterations leads to the isolation of the communities. 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 big activity in the field, and many new methods have been proposed in recent years (see [12] to study these methods). 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.

In comparison with the previous discussions, few works have been done considering the overlapping concept. One of the earliest works in this area was carried out by Baumes et al. which proposed two efficient heuristics, Iterative Scan (IS) and Rank Removal (RaRe) to optimize a given function related to the edge density of the clusters [6]. Gregory proposed CONGA [15] and CONGO [16] based on “split betweenness” to detect overlapping communities in the networks. In 2005, Palla et al. showed that searching for some predetermined structures like fully connected graphs or k-clique in the network can lead to detecting such communities [31]. In another work, Zhang et al. used fuzzy c-means [41] to detect overlapping communities.

Meanwhile very few works are done based on the game theory (see for example [1,9]). 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 [9] 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 [9] is based on the Modularity concept [27] 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.

3. The proposed framework

In this section, first the motivation behind our work and preliminaries are reviewed and then our framework is presented.

3.1. Motivation

As being humans, it is necessary for all of us to make social interactions with each other. None of us like to be alone and we often have incentives to establish friendships with everyone that has a lot in common with us in behavior, ethics, way of life, social interactions, majors, etc. This is the reason that in both real and cyberspace lives, we often belong to communities whose members are very similar to us, including family and friends. This prompted us to consider the community formation problem in social networks as a play of interactions between their constituents, i.e. people.

We have embedded game theory into our work because it provides rigorous mathematical models of strategic interactions between rational, autonomous and intelligent agents. Indeed game theory is a good tool to capture both the behavior of individuals and strategic interactions between rational agent who must decide between operators while in the second phase, it responds to two systematic operators such as join, leave, switch and no operation operators while in the second phase, it responds to two systematic operators as well. These two operators include friend suggestion inspired from social networking websites where we receive suggestions from system and eviction borrowed from real life where there are situations that we might be evicted from groups we belong to. Furthermore, instead of using Modularity function [16], here the utility for each agent is calculated based on Pearson similarity function; the more similar the community members, the more their utilities. This indeed has a logical and sensible interpretation.

3.2. Preliminaries

In this section, we formally formulate our framework. Table 1 shows some of the symbols used in the remaining of the paper and their definitions. Suppose that social network underlying graph \( G = (V, E) \), with \( n = |V| \) vertices (nodes) and \( m = |E| \) edges is presented.

As it is mentioned earlier, we put each vertex down to a rational agent who must decide between operators based on its utility. The set of all feasible communities of the network is denoted by \( \{k\} = \{1, 2, \ldots, n\} \) where \( k \) is polynomial in \( n \), however the number of our final communities may be much smaller than \( k \). In our game-theoretic framework each agent preserves a vector of community labels that it belongs to as its strategies. In other words, the strategy of each agent is denoted by \( s_i \subseteq \{k\} \) and strategy profile \( S \) denotes the set of strategies of all agents, i.e. \( S = (s_1, s_2, \ldots, s_n) \).

Joining to a new community usually is beneficiary for us, while in most cases we face expenses (e.g. fees). Therefore, the utility function \( u_i \) for agent \( i \) can be regarded as a difference between gain and loss functions:

\[
u(s) = g(S) - l(S),
\]

where \( g_i \) and \( l_i \) are gain function and loss function for agent \( i \), respectively:

\[
g_i(S) = \frac{1}{m} \sum_{j \in L,j \neq i} \sum_{j \in L,j \neq i} C_{ij},
\]

\[
l_i(S) = \frac{1}{m} (|s_i| - 1).
\]

Here \( s_i = \{1, 2, \ldots, k\} \) is the labels which agent \( i \) belongs to.

In our framework, the best response strategy of an agent \( i \) with respect to strategies \( S_{-i} \) of other agents is calculated by:

\[
\arg \max_{s'_i \subseteq \{k\}} g_i(S_{-i}, s'_i) - l_i(S_{-i}, s'_i).
\]
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 its current utility:

\[
\forall i, s'_i \neq s_i, \quad u_i(S_{-i}, s'_i) \leq u_i(S_{-i}, s_i). \tag{5}
\]

We used local Nash equilibrium [3,10] in this game, because reaching global one is not feasible. In other words, the strategy profile \( S \) forms a local equilibrium if all agents play their local optimal strategies.

\[
\forall i, s'_i \in \text{Is}(s_i), \quad u_i(S_{-i}, s'_i) \leq u_i(S_{-i}, s_i). \tag{6}
\]

Here \( \text{Is}(s_i) \) refers to local strategy space of agent \( i \), which is the set of possible label sets it can obtain by performing the actions defined in the Table 2 one at a time.

Considering \( G \), one can compute the similarities between each pair of vertices with respect to some local or global properties; no matter whether they are directly connected or not. Many similarity measures exist in the literature that are at the basis of traditional methods like hierarchical, partitional and spectral clustering [12]. These measures are mainly divided into two categories: when it is possible to embed the graph vertices in a Euclidean space, the most prominent measures are Euclidean distance, Manhattan distance and cosine similarity, but when a graph cannot be embedded in space, adjacency relationships between vertices are used [36].

In this work we have used Pearson correlation as a similarity measure which works based on adjacency relationships and structural equivalence [22]. Two vertices are called structural equivalent if they have the same neighbors, even if they are not directly connected. Pearson correlation coefficient is a measure related to structural equivalence which calculates correlation between columns or rows of an adjacency matrix. This measure can be used in order to calculate similarities between each pair of nodes of a graph:

\[
C_{ij} = \frac{1}{n} \sigma_i \sigma_j \sum_{k=1}^{n} (A_{ik} - \mu_i)(A_{jk} - \mu_j), \tag{7}
\]

where \( \mu_i \) is the average and \( \sigma_i \) is the variance:

\[
\mu_i = \frac{1}{n} \sum_{j=1}^{n} A_{ij}, \tag{8}
\]

\[
\sigma_i = \sqrt{\frac{1}{n} \sum_{j=1}^{n} (A_{ij} - \mu_i)^2}. \tag{9}
\]

Pearson correlation metric quantifies how similar are two nodes and is measured from \(-1\) to \(+1\). Score \(1\) indicates that they are perfectly similar and a score of \(-1\) means that they are dissimilar completely. Meanwhile score of \(0\) say nothing about similarity status. Put it simply, the Pearson correlation score, determines how well two data objects fit a line.

### 3.3. Framework

Our proposed framework shown in Algorithm 1, consists of two main phases: **Personal Phase** and **System Phase**. Both of these phases are based on the similarity measure. After calculating similarities between each pair of agents according to (7), the game in the multiagent environment is started. During the game, the actions shown in Table 2 are chosen by each agent and system. The overlapping community structure of the network comes into the view after agents reach local equilibrium as mentioned in the previous section.

In this study, we have assumed that the underlying graph of a given social network is undirected and unweighted, although it is straightforward to extend it to handle directed and weighted graphs as well. The details of our framework and its algorithmic description are described in the followings.

#### 3.3.1. Personal phase

In this phase, an agent is selected randomly from the pool of agents. This agent, periodically makes personal decisions in order to gain better utility while it is in the game. Specifically, it decides whether to join to a new community \( C \) by adding its label to \( s_i \) and gains utility.

<table>
<thead>
<tr>
<th>Action</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Join</td>
<td>Add a new label to ( s_i )</td>
</tr>
<tr>
<td>Leave</td>
<td>Remove a label from ( s_i )</td>
</tr>
<tr>
<td>Switch</td>
<td>Remove a label from ( s_i ) and add a new one</td>
</tr>
<tr>
<td>Suggestion</td>
<td>Suggest a friend to a community</td>
</tr>
<tr>
<td>Eviction</td>
<td>Remove a member from a community</td>
</tr>
<tr>
<td>No Action</td>
<td>No specific action is performed</td>
</tr>
</tbody>
</table>
Algorithm 1. The proposed method

1: \textbf{Input:} underlying network graph $G$.
2: \textbf{Output:} communities as a final division of $G$.
3: \textbf{communities} = \{\}
4: Agents = \{agent$_1$, agent$_2$, ..., agent$_n$\}.
5: Similarity\_Calculate(Agents).
6: \textbf{repeat}
7: \texttt{/* Personal Phase */}
8: action$_i$ = Random\_Select(Agents).
9: $u'_i$ = Utility\_Calculate(agent$_i$, action$_i$).
10: \textbf{if} ($u'_i < u_i$) \textbf{then}
11: \hspace*{1em} $u_i ← u'_i$.
12: \hspace*{1em} Update $s_i$.
13: \hspace*{1em} Update communities.
14: \textbf{else} action$_i$ = no Operation.
15: \textbf{end if}
16: \texttt{/* System Phase */}
17: \textbf{if} \texttt{(communities} $\neq$ \{\}) \textbf{then}
18: \hspace*{1em} $C$ = Random\_Select(communities).
19: \hspace*{1em} agent$_a$ = Evict($C$).
20: \textbf{if} (agent$_a$ $\neq$ \{\}) \textbf{then}
21: \hspace*{1em} $u_a ← u_{Leave}$.
22: \hspace*{1em} Update $s_a$.
23: \hspace*{1em} Update communities.
24: \textbf{end if}
25: \hspace*{1em} $C$ = Random\_Select(communities).
26: \hspace*{1em} agent$_b$ = Suggest($C$).
27: \textbf{if} (agent$_b$ $\neq$ \{\}) \& $u_b < u_{Join}$ \textbf{then}
28: \hspace*{1em} $u_b ← u_{Join}$.
29: \hspace*{1em} Update $s_b$.
30: \hspace*{1em} Update communities.
31: \textbf{end if}
32: \textbf{until} (local equilibrium is reached)

$u_{Join}$ according to (1):

\begin{equation}
\begin{aligned}
s_i ← s_i ∪ \{C\},
\end{aligned}
\end{equation}

or leave one of its own communities, say $C'$ by removing its label from $s_i$ and gains utility $u_{Leave}$:

\begin{equation}
\begin{aligned}
s_i ← s_i \setminus \{C'\},
\end{aligned}
\end{equation}

or switch from a community, say $C'$ by removing its label from $s_i$ to a new one, $C$, by adding its label to $s_i$ and receives $u_{Switch}$ as its utility:

\begin{equation}
\begin{aligned}
s_i ← s_i \setminus \{C'\}, \quad s_i ← s_i ∪ \{C\}.
\end{aligned}
\end{equation}

Finally, the new utility $u'_i$ for this agent is calculated and its old utility $u_i$ is replaced by this new one:

\begin{equation}
\begin{aligned}
u'_i ← \max\{u_{Join}, u_{Leave}, u_{Switch}\}.
\end{aligned}
\end{equation}

This agent can be indifferent as well and select none of these operations if they do not improve $u_i$.

3.3.2. System phase

It is possible that one community decides to evict one of its useless members by making communal decision. In this way, system chooses a community $C$ from existing communities and tries to evict one of the less profitable members say $a$ of this community if there is any. In this case, the selected agent must leave the group and has no further option:

\begin{equation}
\begin{aligned}
\sum_{i \in C ∪ \{a\}} u_i > \sum_{i \in C} u_i
\end{aligned}
\end{equation}

In addition, based on the friend suggestion systems used in almost all social networking websites, system may choose a random community $C$ from existing communities and send a suggestion to agent $a$ to join to this community. This agent is assumed to be profitable for this community, i.e. the summation of their utilities is boosted and if accepts the suggestion only if it is guaranteed to be of worthy for it, otherwise it rejects it:

\begin{equation}
\begin{aligned}
\sum_{i \in C ∪ \{a\}} u_i > \sum_{i \in C} u_i
\end{aligned}
\end{equation}

4. Experimental settings

Evaluating a community detection algorithm is a very hard job since identifying communities is somehow an art than science [10] and obtaining ground truth community information from real-world networks is really challenging issue. We have conducted our experiments on two broad categories of datasets: synthetic datasets and real world graphs. To be more specific, we have tested our algorithm on two synthetic datasets, namely LFR synthetic Networks and Erdős–Rényi Random Networks, on two well-known real world graphs, Flickr and DBLP and also on three small real world graphs, Dolphin network, Zachary Karate Club network and American College Football network.

We have implemented the framework in JAVA and on a system with Processor Intel® Core™ 2 Duo CPU 2.53 GHz and 4 GB RAM. In the next subsections, we first explain the datasets and evaluation metrics used in
our work and then we discuss our results and observations.

4.1. Synthetic datasets

4.1.1. LFR synthetic networks
We ran our method on a set of benchmark graphs recently proposed by Lancichinetti and Fortunato [19]. Their method to create benchmark graphs has a computational complexity that grows linearly with the number of links and reduce considerably the fluctuations of specific realizations of the graphs so that they come as close as possible to the type of structure described by the input parameters. However, it is rather hard to justify how realistic these benchmark graphs are. The graphs are constructed in the following steps [19]:

1. Generating the number of communities that each node belongs to and then assign the degrees to the nodes based on a power law distribution with exponent $\tau_1$.
2. Assigning the community sizes from another power law distribution with exponent $\tau_2$ for a fixed number of communities.
3. Generating the bipartite graph between the nodes and the communities with the configuration model.
4. Assigning the cross-community degree and internal degrees within each community to each node based on $\mu$ (mixing parameter).
5. Building graphs for each community and the cross-community edges with the configuration model.

4.1.2. Erdős–Réyni random networks
As another synthetic dataset, we conducted an experiment on Erdős–Réyni random networks [20]. This method generates random networks which contain no communities and meaningful relationships between the nodes. Lancichinetti et al., mentioned that it is essential for a community detection algorithm to identify a random network with no community structure [20]. In this experiment, the network takes different sizes of 100, 500 and 1000 and our method considers all nodes as a single community while other competing algorithms detects several small size communities.

4.2. Real world graphs

4.2.1. Flickr
Flickr$^5$ is a content sharing website with a focus on photos and also an online community platform. Users can create profiles, upload their own photos and subscribe to different groups of interests. To create this dataset,$^6$ 195 groups of interests are picked randomly [35] and users with only single connection were removed from the dataset. The statistical properties of Flickr is depicted in Table 3.

4.2.2. DBLP$^7$

The study of co-authorship network in academic community has attracted much attention recently [26]. The DBLPS$^8$ is a collection of bibliographic information on major computer science journals and proceedings, which can be used to test the performance of community detection methods. In this collection, each paper is represented by a bag of words that appeared in the abstract and title of the paper. We used this dataset for author name disambiguation.

4.2.3. Dolphin network

This network consists of 62 nodes standing for bottle nose dolphins and 159 edges demonstrating the relations and interactions between them and is a well-known benchmark to test the performances of different algorithms for community detection. It is mentioned in [23] that this network originally is divided into two separate groups.

4.2.4. Zachary Karate Club network

This is another well-known network which consists of 34 nodes and 78 edges and shows the relationships between Zachary Karate Club members [21]. Figure 1 shows the ground truth of this network with two disjoint communities [28].

4.2.5. American college football network

This network was previously used by Girvan and Newman [14] and its community structure is known. The network contains 115 nodes and 613 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, since it occurs that games held between teams of the same conferences are more than games played between different conferences.

<table>
<thead>
<tr>
<th>#Categories</th>
<th>n</th>
<th>m</th>
<th>Max degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>195</td>
<td>80,513</td>
<td>5,899,882</td>
<td>5,706</td>
</tr>
</tbody>
</table>


$^7$Digital Bibliography and Library Project.

$^8$www.informatik.uni-trier.de/˜ley/db/.
4.3. Evaluation metrics

Checking the performance of a community detection algorithm involves defining a criterion to establish how similar the partition delivered by the algorithm is to the partition one wishes to recover [12]. Several measures exist to figure out the similarity of partitions which can be divided in three broad categories: measures based on pair counting, cluster matching and information theory.

Here, we evaluated our results with respect to three well-defined metrics: normalized mutual information, fraction of correctly classified nodes and modularity. Experimental results demonstrate that our approach outperforms other methods that are capable to detect overlapping communities such as Clique [31] with sizes of 3 to 8 and Game [9]. A brief explanation of these metrics are presented in the subsequent sections.

4.3.1. Normalized mutual information

We adopted the extended version of normalized mutual information (NMI) [21] as a measure of similarity borrowed from the Information Theory concepts. This extended version can also support the overlapping communities and is calculated by:

\[
N(X|Y) = 1 - \frac{1}{2} \left[ H(X|Y)_{\text{norm}} + H(Y|X)_{\text{norm}} \right].
\]  

(16)

This variable is in the range \([0, 1]\) and equals 1 when two partitions \(\zeta'\) and \(\zeta''\) are exactly coincident. To calculate this metric, at first we must calculate the following equation:

\[
H(X|Y)_{\text{norm}} = \frac{1}{|\zeta'|} \sum_k H(X_k|Y)_{\text{norm}},
\]

(17)

where we have:

\[
H(X_k|Y)_{\text{norm}} = \frac{H(X_k|Y)}{H(X_k)},
\]

and:

\[
H(X_k|Y_l)_{\text{norm}} = \min_{l \in \{1, 2, ..., |\zeta''|\}} H(X_k|Y_l),
\]

(19)

\[
H(X_k|Y_l) = H(X_k, Y_l) - H(Y_l).
\]

(20)

In (17)–(20), \(X_k = (X)_k\), \(H(X)\) and \(H(Y)\) are entropies of random variable \(X\) and \(Y\) and \(H(X_k|Y)\) is conditional entropy of \(X_k\) with respect to all the components of \(Y\). Finally \(H(X|Y)_{\text{norm}}\) is the normalized conditional entropy of \(X\) with respect to \(Y\). There are neat expressions on how to calculate NMI in this way, in [21].

4.3.2. Fraction of correctly classified nodes

In order to calculate the fraction of correctly classified nodes, we used the method in [28]. Although this method does not support the overlapping concept, it is relatively appropriate estimator of how good a community detection method is. To calculate this metric, first, we searched for a correspondence with the maximal total number of common members between the communities found and the real ones. In order to make the correspondence a one-to-one mapping, we assigned a real community index to a community found with the largest number of members in common, for the indexed real community and the one found, and each real community index can only be assigned once. We continued this assignment until all communities found had been re-indexed or the indexes of the real communities were used up. Consequently, the accuracy is the sum of the number of common members for the communities found and their corresponding real communities. Assuming that the number of correctly classified nodes for each delivered community is denoted by CCN and the set of delivered communities as \(C\), the fraction of correctly classified nodes (FCCN) is calculated using (21):

\[
\text{FCCN} = \frac{\sum_{i \in C} \text{CCN}_i}{n}.
\]

(21)

4.3.3. Modularity

As it is mentioned before, when the ground truth structure of underlying graph is provided, NMI can be used as a useful evaluation. However, in some special cases, we have to test the algorithm on some synthetic datasets with no ground truth and in this case, modularity is used. Although this measure has drawbacks and
becomes unreliable when our networks are too sparse [13], modularity is the most popular qualitative measure in detecting communities in social networks. The higher this metric, the finer the discovered communities.

Modularity is calculated according to the following equation:

$$Q = \sum_{s=1}^{k} \left[ \frac{l_s}{T} - \left( \frac{d_s}{2L} \right)^2 \right]$$  \hspace{1cm} (22)

where $k$ is the number of communities, $l_s$ is the number of edges in the community $s$, $d_s$ is sum of degrees in the community $s$ and $L$ is the number of total edges in the graph.

4.4. Discussion

4.4.1. Analysis of results

Here, we discuss and analyze our results on different datasets in the order that they appeared in the last section and in terms of metrics that we mentioned before.

The first metric we adopted here, is Normalized Mutual Information. As it is mentioned before, we used the extended version of NMI which works well with both overlapping and non-overlapping communities. We ran the method on LFR synthetic datasets. In order to neglect the effect of randomness in our method, we ran it 100 times on each of the synthetic datasets. The results on these graphs with sizes of 1000 and 10,000 nodes and in term of NMI are shown in Figs 2 and 3 respectively. In these figures, the $y$-axis is NMI and the $x$-axis shows different fractions of overlapping nodes of benchmark graphs from 0 to 0.5, i.e., the fraction of nodes that are present in more than one community simultaneously to all nodes. Furthermore, $s_{\text{min}}$ and $s_{\text{max}}$ refer to the minimum and maximum community sizes respectively. The remaining parameter settings are as the following: $\tau_1 = 2$, $\tau_2 = 1$, $k_{\text{avg}} = 20$, $k_{\text{max}} = 50$, $\text{om} = 2$, $\mu = 0.1$ and $\mu = 0.3$ ($\mu$ is a proportion of crossing edges and is called a mixing parameter).

As another evaluation, to show that the proposed method performs well with graphs with no community structure, we ran the method on Erdős–Réyni ran-
Fig. 3. Comparative evaluation of the performances of different algorithms in terms of average NMI, on benchmarks with 10000 nodes. In (a) and (c) $s_{\text{min}} = 20$ and $s_{\text{max}} = 100$. In (b) and (d) $s_{\text{min}} = 10$ and $s_{\text{max}} = 50$. In (a) and (b), $\mu = 0.3$ and in (c) and (d) $\mu = 0.1$. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-130557.)

Table 4

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (s)</th>
<th>#Communities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our method</td>
<td>53,570</td>
<td>128,591</td>
</tr>
<tr>
<td>Game</td>
<td>36,029</td>
<td>202,171</td>
</tr>
<tr>
<td>Clique</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

Computational time of three methods and the number of extracted communities related to Flickr dataset.

In random networks, no meaningful communities were detected and this shows that our method performs well with graphs with no community structure. We also ran our method on real world datasets which is described here. In the case of the Flickr dataset, since there is no ground truth results about the number and structure of the communities, in Table 4, we only shows the running time of three method on this dataset and also the number of extracted communities by them. As we can see, the Game method usually tends to detect all possible communities even those inside the other already found communities, which is not desirable in most of the times. Another observation is that Clique method cannot reach to the solution, since the graph is too large for it. This relatively high time complexity for both applicable methods is due to the high number of nodes and links of the Flickr graph.

As an application, we used a subset of authors of the DBLP records in author name disambiguation. Name ambiguity can affect the quality of scientific data gathering, can decrease the performance of information retrieval and web search, and can cause the incorrect identification of and credit attribution to authors. Hence, proposing methods to solve the name ambiguity problem is interesting to many researchers. Here, we first constructed the co-authorship graph in which each node stands for a name and may represent more than one person simultaneously and then ran our method on this graph. We selected papers from DBLP authored by people whose family names were ‘Lee’ and ran our method on this subset. Five communities out of 15 communities found by our method are shown in Table 5. The different communities show that when searching for a family name ‘Lee’, our method is able to extract different persons with this family name.
The best results of our algorithm with maximum possible NMI, i.e., 1.00, on Dolphin network, Zachary Karate Club network and American College Football network are shown in Figs 4 and 5, respectively. As we can see from these figures, for Dolphin and Zachary network, 2 communities and for Football network, 12 communities are detected by our method. On the other hand, as it is shown in Fig. 5 and Table 6, Game method finds more than two communities for first two networks and 12 teams for the last one. We see that although these networks have not overlapping communities in real, the Game method mistakenly detects overlapping communities. The average results of 100 runs on these graphs are also shown in Table 7.

The next metric, FCCN, shows the fraction of nodes that are classified in their correct communities. The average results in terms of this metric on real world graphs are shown in Table 7. Like before, the method shows promising results on these networks. Additionally, the average results on LFR benchmark graphs are depicted in Fig. 6. This figure shows the superiority of our method compared with two rival methods.

As we discussed before, the former metrics are usually used when the ground truth of the underlying network is available. However, in most of the times, this is not applicable. In this case, modularity can be used. The average results in terms of this metric are shown in Table 7. Again, the proposed method performs better than the other methods since it has higher modularities.

### 4.4.2. Analysis of time complexity

According to Algorithm 1, it is obvious that the personal phase needs $O(k_1 \cdot c_1 \cdot d)$ operations while the system phase requires $O(k_2 \cdot c_2 \cdot D)$ operations to finish. Here, $k_1$ and $k_2$ are constants, $c_1$ and $c_2$ refers to the average times that each agent and community are selected respectively, $d$ is the average degree of nodes and $D$ is the average sum of degrees of the selected community’s members.

It is worthy to note that since calculating the similarities between agents is carried out in an offline mode and before the main parts of our method begin, we do not take into account the time that is necessary to accomplish this task. The total order of our algorithm is therefore sum of the order of two main parts, $O(k_1 \cdot C_1 \cdot d + k_2 \cdot C_2 \cdot D)$. This indeed endorses the most prevalent drawback of our method. Though we reach to the finer and more accurate results compared to the other methods, our running time and number of performed actions are relatively high. In Fig. 7, we observe that our method has higher number of performed actions and hence higher running time shown in Fig. 8 in comparison with its main rival method, i.e., Game method. This is due to the existence of embedded communal decisions in our method, that despite their help in detecting finer communities, result in growing our running time considerably.

### 5. Conclusion and future work

Taking inspiration from human–human interactions, we proposed a game-theoretic framework to identify overlapping communities in social networks based on structural equivalence relations.

Considering the graph of underlying social network as a conceptual social networking website, we attributed community detection problem to an iterative game between the vertices which are regarded as rational agents. Due to the very large strategy space, reaching to the global Nash equilibrium of this game is not viable, so we are just satisfied by a local one. This framework consists of two phases: Personal and System phases where in the first phase, each agent personally decides between join, leave, switch and No Action operations and in the second phase, system decides to suggest a friend to an agent or to evict an agent from an existing community. Regarding to the represented theoretical basis of our method in the Appendix and based on the observations from experiments, it has promising results on the benchmark graphs compared to the other similar methods at the expense of high running time and performed actions.

The framework could be easily extended to use in directed, weighted, multi-dimensional and dynamic networks. Future works include embedding other similar operators borrowed from real life or using some other similarity methods instead of Pearson correlation.
Fig. 4. The best community structure of Dolphin and Zachary networks with maximum possible NMI. (a), (b) Reprinted from [16]. (c), (d) Discovered by our method. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-130557.)
Fig. 5. The best community structure of American College Football network found by our method with NMI = 1. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-130557.)

Table 6
The best community structure of Football network found by Game method

<table>
<thead>
<tr>
<th></th>
<th>1 0, 24, 32, 36, 44, 61, 79, 88, 102, 104, 108, 114</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1, 2, 4, 9, 10, 39, 51, 71, 73, 80, 83, 96, 97, 101, 106, 114</td>
</tr>
<tr>
<td>3</td>
<td>1, 5, 12, 14, 31, 38, 46, 59, 63, 99, 105, 114</td>
</tr>
<tr>
<td>4</td>
<td>3, 8, 15, 22, 40, 66, 92, 103, 114</td>
</tr>
<tr>
<td>5</td>
<td>3, 4, 10, 23, 27, 49, 68, 89, 114</td>
</tr>
<tr>
<td>6</td>
<td>6, 7, 8, 20, 21, 22, 49, 50, 67, 76, 77, 107, 110, 114</td>
</tr>
<tr>
<td>7</td>
<td>11, 13, 17, 25, 30, 33, 35, 37, 41, 42, 53, 60, 70, 84, 98, 114</td>
</tr>
<tr>
<td>8</td>
<td>16, 19, 26, 35, 55, 57, 58, 61, 62, 64, 65, 69, 75, 86, 94, 95, 96, 112, 114</td>
</tr>
<tr>
<td>9</td>
<td>18, 28, 29, 34, 43, 54, 78, 79, 81, 90, 92, 93, 100, 114</td>
</tr>
<tr>
<td>10</td>
<td>23, 45, 48, 52, 66, 72, 82, 83, 87, 109, 114</td>
</tr>
<tr>
<td>11</td>
<td>43, 47, 56, 65, 74, 85, 90, 91, 95, 96, 111, 114</td>
</tr>
<tr>
<td>12</td>
<td>113, 114</td>
</tr>
</tbody>
</table>

Table 7
Average NMI, FCCN and Q on real world graphs by three methods

<table>
<thead>
<tr>
<th>Network</th>
<th>NMI</th>
<th>FCCN</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dolphin</td>
<td>0.633</td>
<td>0.779</td>
<td>0.561</td>
</tr>
<tr>
<td>Karate</td>
<td>0.301</td>
<td>1.00</td>
<td>0.253</td>
</tr>
<tr>
<td>Football</td>
<td>0.887</td>
<td>0.928</td>
<td>0.716</td>
</tr>
</tbody>
</table>

Acknowledgement

This work is supported by Iranian Telecommunication Research Center (ITRC) under Grant No. T/500/13226.

Appendix

In this section, we prove the existence of the Nash equilibrium by presenting definitions, theorems and their proofs. As in “matching pennies” game [30],
some games don’t have Nash Equilibrium. To see when a certain game has Nash equilibrium, recall that potential games are a general class of games that permit pure Nash equilibrium [29]. Indeed for any finite game, there exists a potential function \( \Theta \) defined on the strategy profile \( S \) of the agents that maps this profile to some real values. This function must validate the following condition:

\[
\forall i, \quad \Theta(S) - \Theta(S_{-i}, s'_i) = u_i(S_{-i}, s'_i) - u_i(S).
\] (23)

Equivalently, if the current strategy profile of the game is \( S \) and the agent \( i \) switches from strategy \( s_i \) to \( s'_i \), the potential function exactly mirrors the changes in the agent utility. It is not hard to see that a game has at most one potential function. A game that does possess a potential function is called a potential game. Consequently we have the following theorem.

**Theorem 1.** Every potential game has at least one pure Nash equilibrium, namely the strategy profile \( S \) that minimizes \( \Theta(S) \) [29].

**Proof.** Let \( \Theta \) be a potential function for this game and let \( S \) be a pure strategy profile minimizing \( \Theta(S) \). Consider any action performed by player \( i \) that results in a new strategy profile \( S' \). By assumption, \( \Theta(S') \geq \Theta(S) \) and by the definition of a potential function, \( u_i(S') - u_i(S) = \Theta(S) - \Theta(S') \). Thus utility of agent \( i \) cannot increase from this move and hence \( S \) is stable [9]. □

Now we will provide a sufficient condition to prove our community formation game as a potential game and thus address the existence of the Nash equilibrium. First we have following definition [9].

**Definition (Locally linear function).** A set of functions \( \{f_i, 1 \leq i \leq n\} \) is locally linear with locally factor \( \rho \) if for every strategy profile \( S \) the following con-
Theorem 2. Let \( \{g_i, 1 \leq i \leq n\} \) and \( \{l_i, 1 \leq i \leq n\} \) be the sets of gain and loss functions of a community formation game. If these sets are locally linear functions with linear factors \( \rho_G \) and \( \rho_L \), then the community formation game is a potential game [29].

**Proof.** We define a potential function as \( \Theta(S) = \rho_l(S) - \rho_g(S) \) and assume that agent \( i \) who changes its strategy from \( s_i \) to \( s'_i \). Based on the definitions of locally linear functions and the utility functions \( u_i(\cdot) \), we have \( \Theta(S) - \Theta(S_{-i}, s'_i) = u_i(S_{-i}, s'_i) - u_i(S) \). Therefore, the community formation game is a potential game [9]. \( \square \)

In the followings, we will show that our gain and loss functions are locally linear by demonstrating that they are valid in (23). At first, suppose that agent \( a \) decides to change its strategy from \( s_a \) to \( s'_a \). From the left hand side of (24) we have:

\[
\begin{align*}
g_a(S_{-a}, s'_a) - g_a(S) &= \frac{1}{m} \sum_{A \in s'_a} \sum_{j \in A, j \neq a} C_{aj} \\
 &= \frac{1}{m} \sum_{A \in s_a} \sum_{j \in A, j \neq a} C_{aj} \\
 &= \frac{1}{m} C_{ab}.
\end{align*}
\]

From the right-hand side of (24), we have the following equation:

\[
\begin{align*}
g(S_{-i}, s'_i) - g(S) &= \sum_{i \in [n]} g_i(S_{-i}, s'_i) - \sum_{i \in [n]} g_i(S) \\
 &= \sum_{i \in [n-1], i \neq a,b} g_i(S_{-i}) + g_a(S_{-a}, s'_a) + g_b(S_{-b}, s'_b) - \sum_{i \in [n-1], i \neq a,b} g_i(S) - g_a(S_{-a}, s'_a) - g_b(S_{-b}, s'_b) - g_b(S).
\end{align*}
\]

And from (25) and (26) we have the followings:

\[
\begin{align*}
g(S_{-i}, s'_i) - g(S) &= \frac{C_{ab}}{m} + \frac{C_{ba}}{m} - \frac{2C_{ab}}{m}.
\end{align*}
\]

Multiplying each side of (27) in \( \rho_g \) we have:

\[
\rho_g(g(S_{-i}, s'_i) - g(S)) = \frac{2\rho_g C_{ab}}{m}.
\]

Comparing (25) and (28), by defining \( \rho_g = 1/2 \) we will reach to the following equation and thereby proving (11):

\[
\begin{align*}
g_i(S_{-i}, s'_i) - g_i(S) &= \frac{1}{2} (g(S_{-i}, s'_i) - g(S)).
\end{align*}
\]

Here, we proved that our gain function is locally linear. Accordingly, the proof of locally linearity of loss function is as follows:

\[
\begin{align*}
l_a(S_{-a}, s'_a) - l_a(S) &= \frac{1}{m} (|s'_a| - 1) - \frac{1}{m} (|s_a| - 1) \\
 &= \frac{1}{m}.
\end{align*}
\]

Simply by using \( l \) instead of \( g \) in (26), we have:

\[
\begin{align*}
l(S_{-i}, s'_i) - l(S) &= l_a(S_{-a}, s'_a) - l_a(S) + l_b(S_{-b}, s'_b) - l_b(S) \\
 &= \frac{1}{m} + 0 = \frac{1}{m}.
\end{align*}
\]

Therefore we have:

\[
\begin{align*}
\rho_l(l(S_{-i}, s'_i) - l(S)) &= \frac{\rho_l}{m}.
\end{align*}
\]
Comparing (30) and (32), by defining $\rho_l = 1$ we conclude (23) is satisfied by our loss function. Therefore, this function is locally linear as well.

$$l_i(S_{s-i}, s'_i) - l_i(S) = l_i(S_{s-i}, s'_{s-i}) - l(S).$$ (33)

Here we proved that our gain and loss functions are both locally linear with $\rho_l = \frac{1}{4}$ and $\rho = 1$. So, following Theorem 2, we find that our functions are potential functions and consequently based on Theorem 1, we conclude the proposed framework has Nash equilibrium.

References

International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2009, pp. 817–826.


