Physica A 461 (2016) 840–853

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Using a two-phase evolutionary framework to select multiple network spreaders based on community structure

Yu-Hsiang Fu [a], Chung-Yuan Huang [b,*], Chuen-Tsai Sun [a]

[a] Department of Computer Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 300, Taiwan

[b] Department of Computer Science and Information Engineering, Chang Gung University, 259 Wen Hwa 1st Road, Taoyuan 333, Taiwan

HIGHLIGHTS

- Network community structures are used to identify multiple network spreaders.
- A two-phase evolutionary framework is proposed for finding community structures.
- Our framework produced satisfactory community quality without performance loss.
- A community center measure for selecting network spreaders is described.
- Distinct community structure benefits multi-network spreader dissemination.

ARTICLE INFO

Article history:
Received 26 January 2016
Received in revised form 18 May 2016
Available online 16 June 2016

Keywords:
Genetic algorithm
Community detection
Network spreading
Social network analysis
Multiple network spreaders

ABSTRACT

Using network community structures to identify multiple influential spreaders is an appropriate method for analyzing the dissemination of information, ideas and infectious diseases. For example, data on spreaders selected from groups of customers who make similar purchases may be used to advertise products and to optimize limited resource allocation. Other examples include community detection approaches aimed at identifying structures and groups in social or complex networks. However, determining the number of communities in a network remains a challenge. In this paper we describe our proposal for a two-phase evolutionary framework (TPEF) for determining community numbers and maximizing community modularity. Lancichinetti–Fortunato–Radicchi benchmark networks were used to test our proposed method and to analyze execution time, community structure quality, convergence, and the network spreading effect. Results indicate that our proposed TPEF generates satisfactory levels of community quality and convergence. They also suggest a need for an index, mechanism or sampling technique to determine whether a community detection approach should be used for selecting multiple network spreaders.

1. Introduction

Influential network spreader identification can be used to accelerate or hinder the spread of information, ideas, or diseases in social and complex networks [1–5]. Strategies tied to identified spreaders can be used for tasks such as increasing product exposure ranges in marketing, monitoring contagious disease outbreaks, designing and executing disease

* Corresponding author.
E-mail addresses: yuhsiangfu.cs98g@nctu.edu.tw (Y.-H. Fu), gscott@mail.cgu.edu.tw (C.-Y. Huang), ctsun@nctu.edu.tw (C.-T. Sun).

http://dx.doi.org/10.1016/j.physa.2016.06.042
0378-4371/

intervention strategies [6], and speeding up Internet information dissemination or diffusion [2,4,5]. Thus, researchers are investing considerable time and resources into formulating influential network spreader identification schemes.

In some social network analyses, centrality measures and network structures are used to evaluate node importance. These measures can be classified as *local* or *global* [7–9]. Degree centrality is considered a simple and effective tool for measuring node importance: nodes with high degrees of centrality (e.g., hub nodes with multiple connections) have been shown to exert greater network influence [7–9]. In contrast, global centralities can be measured in terms of betweenness, closeness, and k-shell decomposition [1,3,7–9]. High betweenness indicates the location of a node on many of the shortest communication paths; high closeness indicates that a node serves as a network center with shorter average lengths of the shortest paths to other nodes in the network. High k-shell values (determined using k-shell decomposition) indicate node location within core network layers. A diffusion-based approach such as the PageRank (PR) algorithm can also be used to measure node importance in a network [10]. High PR value indicates node connections with many important neighbors. One research team has identified a method for measuring global diversity and local features that they describe as robust and less sensitive to the effects of network spreading [7,8].

However, centrality methods are insufficient for identifying multiple network spreaders, and are therefore not considered appropriate solutions for many practical marketing and disease prevention applications. Since community structure has been identified as an important network property along with small-world [11], scale-free [12], and fractal properties [13,14], community detection approaches are increasingly being used to identify network community structures or social groups to select multiple network spreaders [9,15–21]. Borgatti has suggested the use of two methods to identify important network nodes for purposes of defining a key-players problem (KPP) for selecting multiple network spreaders: KPP-Pos, a set of key players that are maximally connected to all others nodes, and KPP-Neg, a set of key players that are removed, thereby producing a residual network that has the lowest possible cohesion [22].

According to Borgatti's KPP-Pos definition, finding a set (or simply a limited number) of key players can be reduced to a set/vertex cover problem or a 0/1 Knapsack problem. For successful community detection, the primary challenge is identifying a network's community structure, which in turn supports the identification of multiple network spreaders in communities that are considered representative. Accordingly, a network community detection problem can be reduced to a graph partition problem, but the reduced problem is NP-Complete (NPC) [23]. Evolutionary computing approaches such as genetic algorithms (GAs) [24–29], ant colony optimization (ACO) [30], and particle swarm optimization (PSO) [31] are suitable for finding approximate solutions to NPC problems. GAs have at least two advantages: approximate solutions can be efficiently identified according to specific limitations, and the content of approximate solutions can be analyzed, understood and preserved—that is, community structure information can be extracted from chromosomes and stored in files.

Borgatti's KPP-Neg definition is connected with the problem of finding a set of key players via mapping onto an optimal percolation problem, which minimizes many-body system energy [32]. In optimal percolation, a collective influence (CI) algorithm is used to identify a minimal (optimal) set of key players (influences) that, if removed, could break down an entire network into small, disconnected components at a certain threshold. Weak nodes with small numbers of connections, surrounded by hierarchical hub coronas, emerge from identified optimal sets of influencers in both theoretical and real-world networks. If weak nodes that are capable of spreading information throughout a network are identified as activated or immune, that increases the potential for reducing the number of vaccines that must be distributed and used during a large-scale pandemic. The use of weak nodes is also considered better than centrality-based methods for identifying approximate optimal critical thresholds.

In the present study, network community structures are used to select K network spreaders (or K key players) based on a combination of the KPP-Pos definition, resource limitations (i.e., the number of network spreaders), and community detection. The main criterion for identifying community structures is maximizing modularity [9,15–17,21]. Further, the primary condition for selecting network spreaders is that the nodes must have the highest numbers of intra-community connections and lowest numbers of inter-community connections [22]. According to Kitsak [3], a secondary condition is that distances between spreaders must be considered to determine the extent of network spreading overlap. Selected network spreaders are expected to spread information, ideas, or viruses from inside to outside communities in a network. However, since determining the K number of communities is a difficult task, K usually remains unknown. We therefore propose a *two-phase evolutionary framework* (TPEF) as a trade-off between performance efficiency and community compactness for automatically determining K. In phase one, an appropriate K value is automatically determined based on network topology sampling. In phase two the focus is on optimizing the phase one network partition. Multiple network spreaders are chosen from the identified communities.

For our preliminary experiment, we used the LFR benchmark model [33,34] to perform tests involving a partition-based straightforward GA (SGA) [29], a locus-based GA (LGA) [35–37], and our proposed TPEF to compare execution times, community structure quality, and solution convergence. Results indicate that TPEF is capable of determining an appropriate number of communities (i.e., K number) and generating satisfactory community structures. According to our network spreading simulation results, in the distinct community structure case (i.e., $u < 0.2$) we found that using a community detection approach to select K network spreaders resulted in a much wider spreading range than the centrality-based methods. However, in cases of indistinct community structures (i.e., $u \geq 0.2$), good performance was noted when centrality-based methods were used to select K network spreaders, with results similar to those produced by the community detection approach. In other words, there is at least one benefit to applying community detection methods to assist with K network spreader selection in situations marked by distinct community structures, otherwise centrality-based methods can be used

for K network spreader selection. Our results also indicate that an index, mechanism, or sampling technique is required to determine whether a community detection approach should be applied in order to gain the benefits of community structures.

2. Background

To represent a social/complex network, let an undirected graph $G = (V, E)$, where V is the node set and E the edge set of a network. $|V|$ denotes the number of nodes and $|E|$ the number of edges. The network structure is represented as an adjacency matrix $A = \{a_{ij}\}$ and $a_{ij} \in R^n$, where $a_{ij} = 1$ if a link exists between nodes i and j, otherwise $a_{ij} = 0$.

2.1. Local and global centralities

Degree centrality is a simple yet effective method for measuring node influence in a network [7–9]. Let $C_d(i)$ denote the degree centrality of node i. A high degree centrality indicates a large number of connections between a node and its neighbors. $NB_h(i)$ denotes the set of neighbors of node i at a h-hop distance. The degree centrality of node i is defined as

$$C_d(i) = |NB_h(i)| = \sum_{j=1}^{n} a_{ij} \tag{1}$$

where $|NB_h(i)|$ is the number of neighbors of node i at the h-hop distance; in most cases, $h = 1$.

Betweenness centrality measures the proportion of the shortest paths going through a node in a network [7–9]. Let $C_b(i)$ denote the betweenness centrality of node i. A high betweenness value indicates that a node is located along an important communication path. Accordingly, the betweenness centrality of node i is defined as

$$C_b(i) = \sum_{s \neq t \neq v \in V} \frac{|Q_{st}(i)|}{|Q_{st}|} \tag{2}$$

where $|Q_{st}(i)|$ is the number of shortest paths from node s to node t through node i, and $|Q_{st}|$ the total number of shortest paths from node s to node t.

Closeness centrality measures the average length of the shortest paths from one node to other nodes [7–9]. Let $C_l(i)$ denote the closeness centrality of node i. A high closeness centrality value indicates that a node is located in the center of a network, and that the average distance from that node to other nodes is shorter compared to nodes with low closeness centrality values. The closeness centrality of node i is defined as

$$C_l(i) = \frac{1}{\bar{l}_i}, \quad \bar{l}_i = \frac{1}{n} \cdot \sum_{j=1}^{n} l_{ij} \tag{3}$$

where $\bar{l}_i$ is the average length of the shortest paths from node i to other nodes, and l_{ij} the distance from node i to node j.

2.2. K-shell decomposition and pagerank

K-shell decomposition iteratively assigns a k-shell index value to every node in a network [1,3]. During the first step let $k = 1$, and remove all nodes where $C_d(i) = k = 1$. After removal, the degrees of some remaining nodes may be $k = 1$. Nodes are continuously pruned from the network until there are no $k = 1$ nodes. All removed nodes are assigned a k-shell value of $ks = 1$. This procedure continues until all network nodes are removed and assigned a k-shell index value.

PageRank (PR) is a diffusion-based measure for evaluating node importance and describing a random walk process in a network structure; it is also part of an eigenvector centrality method [4,5,10]. PR was initially created to rank web pages—a high PR value indicates that a web page is connected to other pages that also have high PR values. The PR value of node i is defined as

$$PR(i) = \frac{1-d}{n} + d \cdot \sum_{j \in NB_{h=1}(i)} \frac{PR(j)}{C_d^{out}(j)} \tag{4}$$

where $PR(i)$ is the PR value of node i, $C_d^{out}(j)$ the out-degree of neighbor j, and d a damping factor indicating the probability of a random walker following the link structure. The probability of a random walker randomly jumping to other web pages is $1 - d$. The value of d is usually assigned as 0.85, as is the case in this study.

2.3. Community detection and modularity

The goal in community detection is to identify the best network partition in which edge connectivity is tight inside a community and lose between communities [9,15–21]. Network type variation results in different meanings—for example,

Table 1
Uniform crossover example.

v_i		idv_x		idv_y		idv_{new}
1	→	1	→	1	→	1
2		3		2	→	2
3	→	1	→	2	→	1
4		1		3	→	3
5	→	2	→	3	→	2

in terms of community structure corresponding to a set of web pages on the same topic; of a social group in an acquaintance network; of a circuit, pathway or motif serving a certain synthesizing or regulating function in a metabolic network; or of a target group of customers exhibiting the same purchase behaviors in a social network.

Since there are many possible partitions for any given network, how to measure network partition quality is an important issue. Girvan and Newman have proposed using edge betweenness-based hierarchical clustering to identify community structure [15], as well as a modularity measure to evaluate community structure quality [16,17]. Regarding modularity, a meaningful network partition is one with many edges inside communities and only a few between communities. For a network with K communities, modularity is defined as

$$Q = \sum_{i=1}^{K} \left(e_{ii} - a_i^2\right) = \sum_{i=1}^{K} \left(\frac{p_i}{|E|} - \left(\frac{q_i}{2\,|E|} \right)^2 \right) \tag{5}$$

where e_{ii} is the proportion of edges with two endpoints within community m_i, a_i the proportion of edges with at least one endpoint within community m_i, p_i the number of edges with two endpoints within community m_i, and q_i the sum of the degree of community m_i as a node.

2.4. Community detection using GAs

Multiple network spreader or K key-player selection and community detection problems can be reduced to NPC problems [23]. Evolutionary computing methods such as genetic algorithms (GAs) are generally suitable for finding approximate rather than optimal solutions. Further, single-objective (e.g. Meme-Net) and multiple-objective GAs (e.g.. MOEA/D-Net, GA-Net, MOGA-Net, MIGA, and APMOEA) have been applied to community detection problems [24,25,28,35,36,38–44]. Related gene coding representations (e.g., straight-forward and locus-based) and genetic operations have been proposed, with modularity and community scores used as GA fitness functions [25,29,35–37,40–42]. Affinity propagation (AP) [44,45] has recently been used as a pre-processing method to find initial community structure configurations for all individuals, as well as to accelerate population evolution by identifying good starting points, after which straightforward gene representation and single- or multiple-objective evolutionary algorithms can be used to find satisfactory community detection solutions.

GA gene representation consists of encoding and decoding steps (also referred to as genotypes and phenotypes). For example, in a network with K communities, each network partition solution is represented as an individual (or chromosome) $idv = \langle g_1, g_2, \ldots, g_n \rangle$ in population $pop = \{idv_1, idv_2, \ldots, idv_n\}$. In the encoding step of straightforward representation (Fig. 1), each gene is assigned a community id value $g_i \in [1,\ K]$, meaning that node i belongs to community m_i. During the decoding step, community structure is reconstructed by decoding information encoded in an individual, with each node i assigned to community m_i by gene or community id value g_i. In contrast, in locus-based representation encoding steps, each gene is randomly assigned a neighbor id $g_i \in NB_{h=1}(i)$, meaning that an edge exists between nodes i and j. During the decoding step, other transformation approaches such as the Union-Find algorithm and Disjoint-Set data have been used to reconstruct network community structures [23].

According to the genetic operation literature, examples of selection operations include roulette wheel, tournament, truncation, and elitist, among others [28,29]. Crossover and mutation operation decisions depend on the type of gene representation being used. Since traditional single or multi-point crossovers are capable of destroying community structures encoded in an individual, crossover operations in straightforward representations can be one-way or two-way [25,28,43]. The mutation operation utilizes bit-by-bit mutation or self-evolution (SE) methods [25], compared to locus-based representation in which crossover operations can be uniform [40]. In uniform crossovers, two individuals (idv_x and idv_y) are randomly chosen from a population. A randomly generated mask is used to determine each idv_x or idv_y gene (g_i), which is duplicated in the same position in idv_{new}. Table 1 presents a uniform crossover example. The mutation operation is similar to bit-by-bit, but with the constraint that each gene is randomly assigned a neighbor id based on the mutation rate [35–37,40–42].

3. Proposed framework

The purpose of TPEF is to determine an appropriate number of K communities in a network, as well as performance efficiency and community structure quality. However, efficiency and community structure quality might be considered a

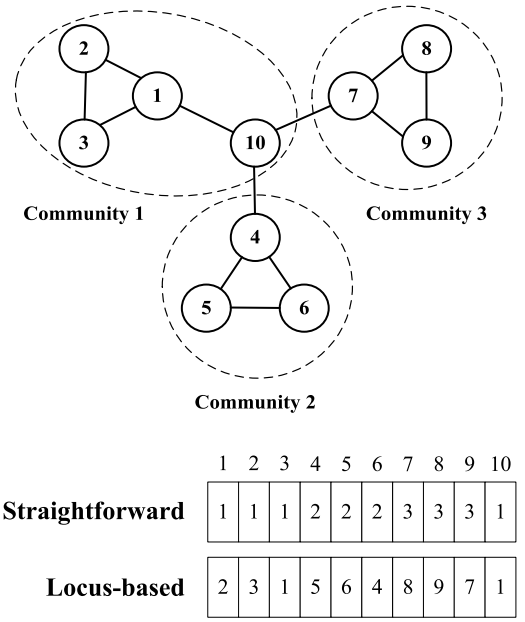


Fig. 1. Straightforward and locus-based gene representations.

trade-off. For a given network with K communities, the gene representation associated with the SGA can result in good performance efficiency, but the large solution search space (i.e., K options for each gene) can result in poor community structure quality. For the LGA, topology-based gene representation can result in a smaller solution search space (i.e., fewer options on average taken from neighbors) and better community structure quality; the K number can be determined automatically. Accordingly, TPEF has advantages over both the SGA and LGA.

The TPEF procedure is shown as Algorithm 1. For a given network, the K number of communities is assumed as unknown. The first step consists of inputting variables for each phase—for example, number of evolution num_{evo}, number of generation num_{gen}, number of population num_{pop}, selection rate $rate_{sel}$, crossover rate $rate_{cro}$ and mutation rate $rate_{mut}$. Next, the LGA is used during phase one to determine a K number with appropriate community structure quality; phase one then returns idv_{best}^{p1}. During phase two, the SGA is used to maximize idv_{best}^{p1} modularity; idv_{best} is determined by comparing the current idv_{best}^{p2} with the previous idv_{best}. Last, TPEF returns the idv_{best} of a given network as the approximate community detection solution.

Algorithm 1 Two-Phase Evolutionary Framework.

Input: network G, number of evolutions num_{evo},
generation of phase1 num_{gen}^{p1}, population size of phase1 num_{pop}^{p1},
selection rate of phase 1 $rate_{sel}^{p1}$, crossover rate of phase1 $rate_{cro}^{p1}$, mutation rate of phase1 $rate_{mut}^{p1}$,
generation of phase2 num_{gen}^{p2}, population size of phase2 num_{pop}^{p2},
selection rate of phase2 $rate_{sel}^{p2}$, crossover rate of phase2 $rate_{cro}^{p2}$, mutation rate of phase2 $rate_{mut}^{p2}$.
Output: best overall evolution idv_{best}.

$idv_{best} \leftarrow \{\}$
For i=1 to num_{evo}:
  $idv_{best}^{p1} \leftarrow Phase1DetermineCommunityNumber(G, num_{gen}^{p1}, num_{pop}^{p1}, rate_{sel}^{p1}, rate_{cro}^{p1}, rate_{mut}^{p1})$
  $idv_{best}^{p2} \leftarrow Phase2MaximizeModularity(G, idv_{best}^{p1}, num_{gen}^{p2}, num_{pop}^{p2}, rate_{sel}^{p2}, rate_{cro}^{p2}, rate_{mut}^{p2})$
  If idv_{best} is empty:
    $idv_{best} \leftarrow idv_{best}^{p2}$
  If idv_{best}^{p2} is better than idv_{best}:
    $idv_{best} \leftarrow idv_{best}^{p2}$
Return idv_{best}

Gene representation, genetic operation, and fitness function configurations for each TPEF phase (as well as SGA and LGA) are shown in Table 2. In TPEF, locus-based representation is used during phase one, with each gene of an individual being randomly assigned by selecting its neighbors' node id. The Union-Find algorithm and Disjoint-Set data structure [23] are used as parts of an additional transformation approach. Straightforward representation is used in phase two, with each gene of an

Table 2
SGA, LGA and TPEF configurations.

Method	SGA	LGA	TPEF	
			Phase1	Phase2
Encoding	Straightforward	Locus-based	Locus-based	Straightforward
Selection	Truncation selection			
Crossover	Uniform crossover			
Mutation	Bit-by-bit mutation			
Fitness	Modularity			

individual being randomly assigned a value picked from the distribution of its neighbor's community id. During truncation selection, the highest chromosome fitness is preserved based on the top $rate_{sel}$ selection rate to create new individuals. In uniform crossover, two parent individuals are randomly chosen from a population, and new offspring are produced by randomly exchanging each gene according to the crossover rate ($rate_{cro}$). Bit-by-bit mutation differs slightly for each TPEF phase: in phase one, each gene mutation is based on the node id of its neighbors; in phase two, each gene mutation is based on the neighbors' community id distribution. In the fitness function, modularity is used and calculated according to Algorithm 2, as in (5).

During each TPEF phase, Algorithm 2 is used to calculate modularity as the fitness value of every individual in the population. First, a community set $M = \{m_1, m_2, \ldots, m_K\}$ is reconstructed using additional transformation approaches to decode the network structure information of an individual; $m_i = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes belonging to community m_i, where $v_j \in V$. An e array of length K is created for the purpose of counting and checking each edge $e_{ij} \in E$ to determine whether the two endpoints are in the same community m_i. An a array is used to accumulate the node degree sum in community m_i. M^{id} is used to store the id of the community that node i belongs to. The next step is to check whether community set M contains an empty set following the mutation operation. Finally, modularity values are calculated as the fitness of each idv, as in (5).

Algorithm 2 Fitness Function.

Input: network G, community M.
Output: fitness value (modularity) of idv.

$Q \leftarrow 0$
$e \leftarrow [0_1, 0_2, \ldots, 0_K]$
$a \leftarrow [0_1, 0_2, \ldots, 0_K]$
$M^{id} \leftarrow [0_1, 0_2, \ldots, 0_{|V|}]$
For m_i in M:
  If m_i is empty:
    Return Q
  For v_j in m_i:
    $a_i \leftarrow a_i + C_d(v_j)$
    $M_j^{id} \leftarrow i$
For (v_s, v_t) in E:
  If $M_s^{id} == M_t^{id}$:
    $e_{M_s^{id}} \leftarrow e_{M_s^{id}} + 1$
For i=1 to K:
  $p_i \leftarrow e_i/|E|$
  $q_i \leftarrow (a_i/2|E|)^2$
  $Q \leftarrow Q + (p_i - q_i)$
Return Q

Multiple network spreader selection is based on TPEF-identified community structures. Each spreader (represented as center T_i and selected from community m_i nodes) is defined as

$$T_i = \arg\max_{v_j \in m_i} C^r_{m_i}(j) \tag{6}$$

$$C^r_{m_i}(j) = \frac{|NB_{h=1}(j) \cap m_i|^2}{C_d(j) \times |m_i|} \tag{7}$$

where $C^r_{m_i}(j)$ is the center degree of node j in community m_i. If $C^r_{m_i}(j) = 1$, node j is connected to all nodes in community m_i; if $C^r_{m_i}(j)$ is close to 0, node j is connected to nodes in other network communities.

Table 3
LFR model parameters.

Parameter	Value
$\|V\|$	1000
γ	2
β	1
u	0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6
$\langle k \rangle$	20
$k_{\max}$	50
$z_{\max}$	50
$z_{\min}$	10

Table 4
Actual/identified community numbers from evolutionary algorithms.

LFR networks									
Mixing parameter u		0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6
K		26	37	37	40	40	40	40	40
$\langle k \rangle$	SGA	26	37	37	40	40	40	40	40
	LGA	26.4	37.8	35.8	33.8	29.8	29.9	32.1	31.3
	TPEF	27	38.3	34.8	34	30.2	28.6	28.4	30.4

Table 5
Evolutionary algorithm parameters.

Method		num_{evo}	num_{gen}	num_{pop}	$rate_{sel}$	$rate_{cro}$	$rate_{mut}$	K
SGA LGA		30	100	50	0.1	0.8	0.05	Known Unknown
TPEF	Phase 1 Phase 2	30	50 50	25 25	0.1	0.8	0.05	Unknown

Table 6
Social network statistics.

Network	$\|V\|$	$\|E\|$	K	$\langle C_c \rangle$	$k_{\max}$	$\langle k \rangle$	r
dolphins	62	159	2	0.259	12	5.129	−0.0436
football	115	613	12	0.4032	12	10.661	0.1624
karate	34	78	2	0.5706	17	4.5882	−0.4756
polblogs	1222	16 714	2	0.3203	351	27.355	−0.2213
polbooks	105	441	3	0.4875	25	8.4	−0.1279
santafe	118	200	4	0.6119	29	3.3898	−0.3075

4. Experimental results and discussion

In our preliminary experiment, the LFR model [33,34] was used to generate synthesized networks with different community structure properties and to test the quality of the community structures identified by the proposed algorithm. Degree and community size were assumed as following power-law distributions in the LFR model, whose parameters were γ, the degree distribution exponent; β, the community size distribution exponent; $k_{\max}$ and $k_{\min}$, the upper and lower node degree boundaries, respectively; $z_{\max}$ and $z_{\min}$, the community size constraints; mixing parameter u, a proportion of a node sharing links with the nodes of other communities; and $1 - u$, a proportion of nodes sharing links with other nodes in the same community. The LFR parameter values [44] used in this study are shown in Table 3. For each u, 10 synthesized networks with the same community number as in Table 4 were generated by the LFR model. Each synthesized network was used for 30 evolutions, and each evolution was run for 50 or 100 generations (Table 5).

Six well-known social networks with ground-truth-community information (i.e., known K values) were used in the preliminary experiment. Chosen social networks were Zachary's karate club [46], dolphins [47], American college football [15], political book [48], political blogs [49], and Santa Fe Institute collaboration [15]. Social network statistics are presented in Table 6, with $|V|$ designating the number of nodes, $|E|$ the number of edges, K the number of communities, $\langle C_c \rangle$ the average clustering coefficient, $k_{\max}$ the maximum node degree, $\langle k \rangle$ the average node degree, and r network assortativity. In our preliminary experiments, each social network was used for 30 evolutions, and each evolution run for 50 or 100 generations.

Normalized mutual information (NMI) was used to measure the quality of community structures identified by the algorithms [50]. The degree of similarity between true partition X and the identified partition Y can be calculated using

NMI, defined as

$$\mathrm{NMI}(X,\ Y) = \frac{2 \cdot I(X, Y)}{H(X) + H(Y)} \tag{8}$$

$$I(X,\ Y) = \sum_{x} \sum_{y} P(x, y) \log \frac{P(x, y)}{P(x)\, P(y)} \tag{9}$$

where $H(X) = -\sum_{x \in X} P(x) \log P(x)$ is the Shannon entropy of true partition X, and $P(x)$ the proportion of community x in partition X. $H(Y)$ is the Shannon entropy of identified partition Y, and $P(y)$ the proportion of community y in partition Y. $I(X,\ Y)$ is mutual information for X and Y indicating the level of correct Y identification by the algorithm when X is given. $P(x,\ y)$ is the percentage of overlap between X and Y. If $\mathrm{NMI}(X,\ Y) = 1$, the partitions are identical, otherwise they are considered independent.

We used a susceptible–infective–recovered (SIR) model and the network-based spreading simulation shown in Algorithm 3 to evaluate the spreading capabilities of multiple network spreaders based on the algorithm's identified community [3,4,7,8,20,51–53]. S set nodes are susceptible to information or diseases, I set nodes are capable of infecting neighbors, and R set nodes are immune and cannot be reinfected. Let $|S_t|$ denote the number of susceptible nodes, $|I_t|$ the number of infected nodes, $|R_t|$ the number of recovered nodes, and ρ_t the proportion of immune nodes—all at time t. The total number of nodes is expressed as $|S_t| + |I_t| + |R_t| = |V|$. During the first step of a network-based spreading simulation, all nodes are in the S state except for the initial spreaders, which are in the I state. During step two, each I state node randomly infects its neighbors according to an infection rate $rate_{inf}$ at each time step t, then enters an R state (i.e., $rate_{rec} = 1$). A cumulative incidence of contagion $\rho_t = |R_t| / |V|$ is calculated at the end of each time step. Step 2 is repeated until the maximum time step requirement is satisfied or, if necessary, when the I state set is empty.

Algorithm 3 Network-based Spreading Simulation.

Input: network G, time-step num_{step}, initial nodes I_{init}, infection rate $rate_{inf}$, recovery rate $rate_{rec}$.
Output: network spreading result ρ.

$S \leftarrow \{\} \cup V$
$I \leftarrow \{\}$
$R \leftarrow \{\}$
$\rho \leftarrow \left[0_0, 0_1, \ldots, 0_{num_{step}}\right]$
For t=0 to num_{step}:
  $I_t \leftarrow \{\}$
  $R_t \leftarrow \{\}$
  IF $t == 0$:
    $I \leftarrow I \cup I_{init}$
    $S \leftarrow S - I$
  Else:
    $I_t \leftarrow SusceptibleToInfected(G, S, I, rate_{inf})$
    $R_t \leftarrow InfectedToRecovered(I, R, rate_{rec})$
    $R \leftarrow R \cup R_t$
    $I \leftarrow I \cup I_t$
    $I \leftarrow I - R_t$
    $S \leftarrow S - I_t$
  $\rho_t \leftarrow |R_t|/|V|$
Return ρ

The hardware used for our experiments consisted of an Intel Core 2 Quad Q8200S @ 2.33 GHz CPU with 4 GB DDR4 RAM (Windows 7 Enterprise 64-bit SP1 OS). SGA, LGA and TPEF programs were constructed using Python 3.4, NetworkX 1.9.1, Numpy 1.9.2 and Scipy 0.15.1. Results were analyzed in terms of execution time, community detection, convergence analysis, and network-based spreading simulation within a multiple network-spreader scenario.

Average run time for each method was calculated over 30 evolutions for each LFR and social network. Evolutionary algorithm settings are shown in Table 5. According to the execution time results shown in Tables 7 and 8, the proposed TPEF outperformed the SGA and LGA in terms of average execution time, since the num_{gen} and num_{pop} parameters were set at one-half those of the SGA and LGA. In some cases performance efficiency is more important than community structure quality, depending on how quickly an appropriate K number of communities is determined. Table 4 presents results for average number of identified communities $\langle K \rangle$ over 30 evolutions, as determined by the three algorithms. When $u = 0.01$–0.1, K can be determined using locus-based gene representation methods such as LGA and TPEF. As shown, when $u = 0.2$–0.6, the identified K moved further away from the actual K as the community structure became increasingly indistinct. Table 4 also illustrates another problem of using locus-based representation for identifying K when community structure is indistinct: community structure becomes fragmented, breaking up into multiple sub-communities. This problem requires a new form of modified locus-based representation, or the application of advanced techniques such as affinity propagation [44,45].

Table 7
LFR network execution time results.

Execution time	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
Mixing parameter	$u = 0.01$		$u = 0.05$		$u = 0.1$		$u = 0.2$	
SGA	75.7525	1.5024	75.3942	0.7579	76.2639	0.7105	73.0285	1.1626
LGA	104.6594	1.5078	105.3658	0.6954	104.7413	0.7852	100.1307	1.2442
TPEF	***45.0691***	***0.8569***	***45.143***	***0.4519***	***44.9456***	***0.3846***	***42.6365***	***0.6726***
Mixing parameter	$u = 0.3$		$u = 0.4$		$u = 0.5$		$u = 0.6$	
SGA	74.5829	0.5295	74.7496	0.5517	74.2955	0.6811	74.6951	0.4876
LGA	100.5619	0.5746	99.7236	0.474	98.9848	0.6394	99.1111	0.6154
TPEF	***42.7698***	***0.3582***	***42.3567***	***0.2825***	***41.9442***	***0.2794***	***42.1601***	***0.2703***

Table 8
Social network execution time results.

Network	dolphins		football		karate		polbooks		polblogs		santafe	
Execution time	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
SGA	1.5990	0.0078	2.7102	0.0113	5.4361	0.1045	4.9494	0.0171	120.1155	***0.325***	4.4502	***0.0127***
LGA	2.3837	0.0144	4.1200	0.0168	8.4053	0.0316	7.3481	0.0341	143.800	0.379	6.9623	0.0255
TPEF	***1.0104***	***0.0077***	***1.7363***	***0.0071***	***3.5849***	***0.0110***	***3.1013***	***0.0123***	***60.8827***	2.0047	***2.8980***	0.0204

Table 9
LFR network modularity results.

Modularity Q	Max.	Avg.	Std.	Max.	Avg.	Std.	Max.	Avg.	Std.	Max.	Avg.	Std.
Mixing parameter	$u = 0.01$			$u = 0.05$			$u = 0.1$			$u = 0.2$		
SGA	0.9235	0.9124	0.0067	0.8367	0.8287	0.0064	0.7196	0.7148	***0.0032***	0.5309	0.5154	0.0086
LGA	0.9463	0.9381	0.0051	0.8819	0.8718	0.0051	0.7871	0.7735	0.0057	0.5996	0.5847	***0.0079***
TPEF	***0.9464***	***0.9427***	***0.0021***	***0.9047***	***0.9000***	***0.004***	***0.8209***	***0.8132***	0.0056	***0.6114***	***0.5946***	0.0101
Mixing parameter	$u = 0.3$			$u = 0.4$			$u = 0.5$			$u = 0.6$		
SGA	0.3540	0.3442	0.0042	0.2456	0.2305	0.0078	0.1595	0.1517	0.0042	0.1075	0.1025	0.003
LGA	***0.4199***	***0.4073***	***0.007***	***0.2771***	***0.2664***	***0.0045***	***0.1899***	***0.1831***	***0.0039***	***0.1339***	***0.1316***	0.0017
TPEF	0.3946	0.3829	0.0062	0.2536	0.244	0.0053	0.1787	0.1667	0.0046	0.1263	0.1241	***0.002***

Table 10
Social network modularity results.

Modularity Q	Max.	Avg.	Std.	Max.	Avg.	Std.	Max.	Avg.	Std.
Network	dolphins			football			karate		
SGA	0.4027	0.3757	0.0357	0.5843	0.5461	0.0161	0.3718	0.3718	***0.0000***
LGA	***0.5285***	***0.5272***	***0.0012***	***0.6046***	***0.6012***	***0.0039***	***0.4198***	***0.4197***	0.0004
TPEF	***0.5285***	0.5257	0.0055	***0.6046***	0.5936	0.0087	***0.4198***	0.4186	0.0044
Network	polbooks			polblogs			santafe		
SGA	0.5221	0.4919	0.0238	***0.4252***	***0.4245***	***0.0003***	0.6869	0.6373	0.0287
LGA	***0.5272***	***0.5272***	***0.0001***	0.4117	0.4039	0.0045	***0.7506***	***0.7505***	***0.0007***
TPEF	***0.5272***	0.5269	0.0011	0.4249	0.4208	0.0027	***0.7506***	0.7497	0.0016

As shown in Table 9, optimum TPEF modularity occurred when $u = 0.01$–0.2, and second best TPEF modularity was observed when $u = 0.3$–0.6 (very close to LGA values). All LGA modularity results exceeded SGA and TPEF when $u = 0.3$–0.6. As shown in Table 10, optimum modularity was identified by LGA and TPEF in all networks except polblogs. On average, LGA modularity results were the best, with TPEF results equal or very close to the LGA and SGA results. Each TPEF phase setting can be adjusted according to individual study requirements or depending on whether the K number is known or unknown. For example, to determine a K number, phase one TPEF parameters can be set close to or at the same values as for the LGA, and to optimize modularity, the phase two TPEF parameters can be set close to or at the same values as for the SGA. To expand TPEF usability, any approach or deterministic method that fits with the purpose of each phase can be combined with TPEF for community detection problems.

Regarding community detection analysis, the SGA was used as a baseline in the case of a known K, and the LGA and TPEF were used for unknown K cases. This allowed for observations of differences between TPEF and SGA/LGA according to different community structures. The NMI was used to evaluate identified community structure quality. As shown in Fig. 2, TPEF retained optimum NMI value when $u = 0.01$–0.1, had an NMI value close to that for the LGA and better than the SGA when $u = 0.2$, and an NMI value that was close to SGA when $u = 0.3$–0.6. Note that SGA and LGA NMI values were identical when $u = 0.3$–0.6. These results suggest that TPEF is capable of identifying appropriate community

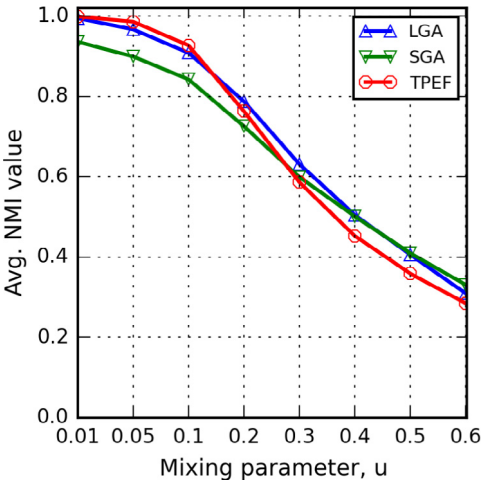
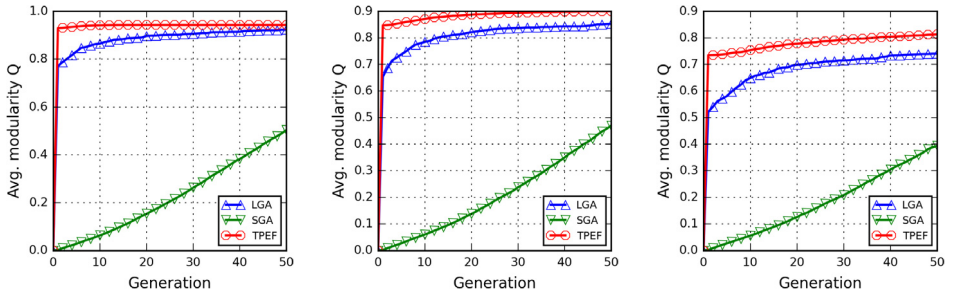
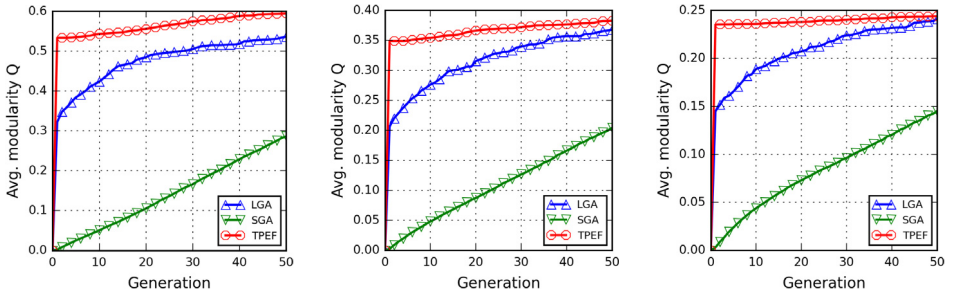


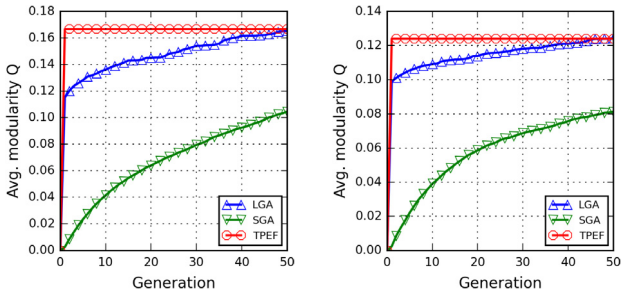
Fig. 2. Normalized mutual information (NMI) results.



(a) $u = 0.01$. (b) $u = 0.05$. (c) $u = 0.1$.

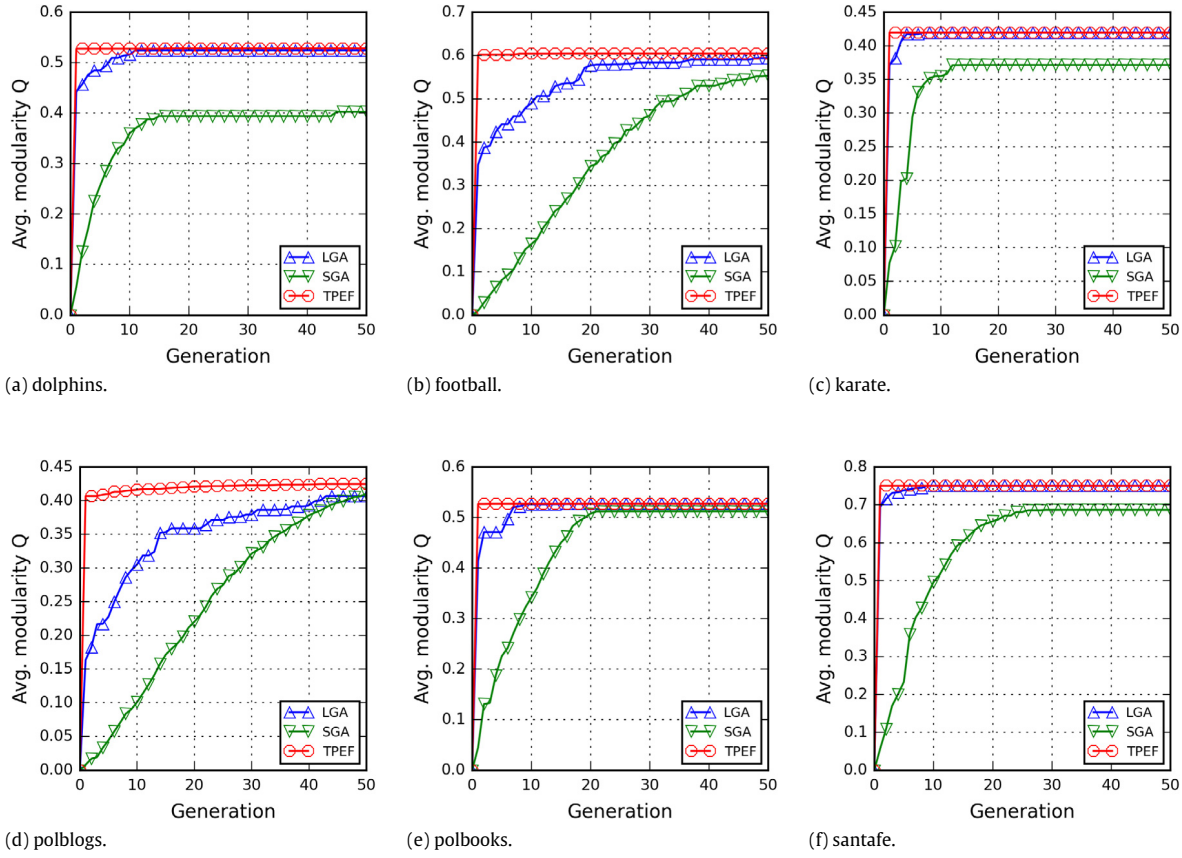


(d) $u = 0.2$. (e) $u = 0.3$. (f) $u = 0.4$.



(g) $u = 0.5$. (h) $u = 0.6$.

Fig. 3. Results from convergence analysis of LFR networks.



(a) dolphins. (b) football. (c) karate.

(d) polblogs. (e) polbooks. (f) santafe.

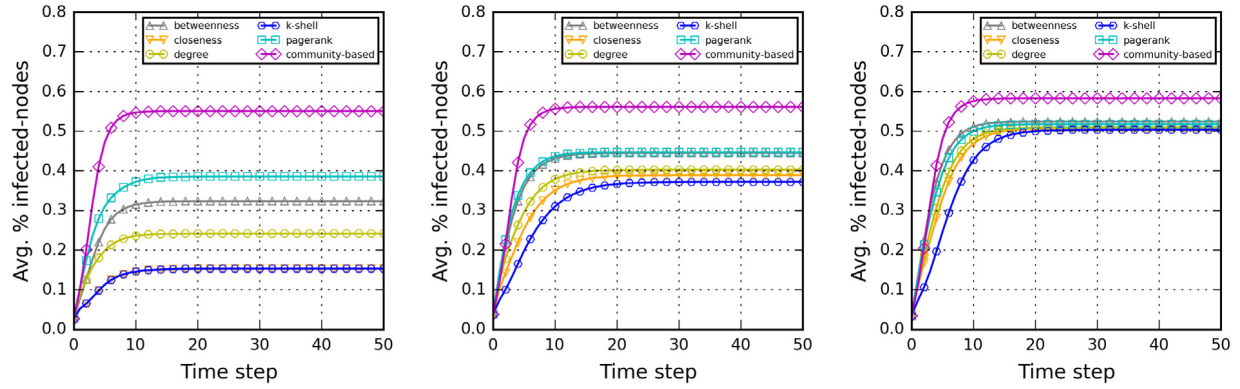
Fig. 4. Results from convergence analysis of social networks.

structures with satisfactory NMI values in cases where the community K number is unknown and community structure is distinct. However, TPEF produced poor results when $u = 0.3$–0.6 due to the trade-off of community structure quality for performance efficiency (Tables 7 and 8). These results can be improved by adjusting the TPEF parameters (e.g. num_{gen} and num_{pop}) to produce better community structure quality without a significant loss of performance efficiency. To make the entire framework more flexible and self-adaptive for different network community structure properties, parameters num_{gen} and num_{pop} can be encoded into individuals and joint evolutionary processes, resulting in a new fitness function that acquires better community structure quality and uses less time for execution.

Fig. 3 presents average convergence data over 30 evolutions for each u of an LFR network. Fig. 3 presents data for the first 50 generations of each method only when TPEF generation number (num_{gen}) = 50 (see also Table 5). For the LFR networks, the TPEF results converged very quickly to an approximate optimal solution at all u values; some final TPEF results were close to LGA results when $u = 0.01$ or 0.3–0.6. As shown in Fig. 4, TPEF results converged more quickly to an approximate optimal solution compared to LGA and SGA, since phase one of TPEF was capable of quickly determining an appropriate community number K, and phase two was capable of continually maximizing modularity.

Last, the network-based spreading simulation and SIR model were used to evaluate the spreading capabilities of multiple network spreaders selected according to different measures and the community-based method, as in (6). A total of 5000 simulations were executed (50 time steps per simulation). ρ_t denotes the average cumulative incidence of contagion—that is, the average number of recovered nodes at time t. In the SIR model, $rate_{inf} = 0.08$ and $rate_{rec} = 1$. According to the multiple network spreader selection strategy, betweenness, closeness, degree, k-shell, and PageRank measures followed the naive top-K strategy, meaning that the top-K nodes were selected as spreaders. The community-based method was used to select representative community centers.

According to the network spreading results shown in Fig. 5, use of the community-based method can result in a higher number of infected nodes compared to the other network measures when $u = 0.01$–0.1; no difference in spreading results was noted when $u = 0.2$–0.6. As shown in Fig. 6, using the community-based method to select multiple network spreaders can result in an approximately 40% gain in network spreading compared to the closeness centrality and k-shell methods when $u = 0.01$; at this point a decrease occurs when u increases. In contrast, when $u \geq 0.2$, using the naive top-K strategy can produce network-spreading results similar to those produced by the community-based method (i.e., benefits less than 5% of the average community effect), while also saving application time.



(a) $u = 0.01$. (b) $u = 0.05$. (c) $u = 0.1$.

Fig. 5. Results for multiple network spreaders in LFR networks with u values of 0.01, 0.05 and 0.1.

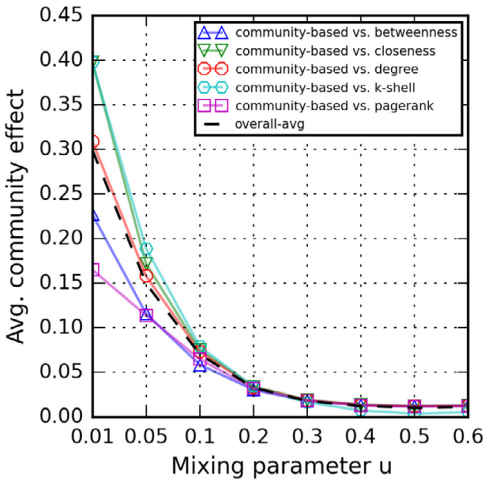


Fig. 6. Benefit of applying the community detection approach.

5. Conclusion

The community detection approach described in this paper can be used to select multiple spreaders in a network. In this study, we assumed an unknown K number of communities when proposing a two-phase evolutionary framework (TPEF) as a trade-off between performance efficiency and community structure quality. In phase one, K is automatically determined according to the network topology structure. In phase two, the modularity of the identified community structure produced by phase one is maximized. Last, multiple network spreaders are selected as community centers based on the identified TPEF community structure.

We used the LFR model to generate synthesized networks with different community structure properties (i.e., different u mixing parameters) and six well-known social networks to test our proposed algorithm. Execution time results indicate that TPEF performed well, with satisfactory community structure quality under specific parameter constraints. According to our community detection analysis, TPEF produced the best NMI-identified community structure values when $u = 0.01$–0.1, and good NMI values when $u = 0.2$. We used network-spreading simulations to compare various measures (e.g., degree centrality) and community-based methods, and to analyze the effects of community structure on network spreading. Our data indicate that using the community detection approach resulted in an average 40% benefit—that is, compared to the results produced by the closeness centrality and k-shell methods, we observed a 40% higher cumulative incidence of contagion in network spreading resulting from the community structure effect when $u = 0.01$.

Our results also suggest that a network's community structure can affect the choice of method when multiple network spreaders are involved. When community structure is obvious (i.e., $u = 0.01$–0.1), community detection approaches can be used to assist in the selection of multiple network spreaders, thus benefitting the network spreading-associated community structure. In contrast, if the community structure is not obvious (i.e., $u \geq 0.2$), the naïve top-K strategy based on clear measurements is sufficient for selecting multiple network spreaders.

This study highlights an interesting research issue: whether community detection approaches should be used to assist in the selection of multiple network spreaders when the K number of communities of a given network is unknown. Our results indicate that community detection approaches require considerable time without providing significant benefits in terms of network spreading when community structure is applied to networks with high u mixing parameters. Accordingly, developing an index, mechanism, or sampling technique is required for determining an appropriate mixing parameter u of a network to support decisions about applying community detection approaches to multiple network spreader selection.

Acknowledgments

The work was supported in part by a Grant from the Republic of China National Science Council (MOST-104-2221-E-182-047). The work was supported in part by the High Speed Intelligent Communication (HSIC) Research Center, Chang Gung University, Taiwan.

References

[1] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, E. Shir, A model of Internet topology using k-shell decomposition, Proc. Natl. Acad. Sci. 104 (2007) 1115011154.
[2] B. Doerr, M. Fouz, T. Friedrich, Why rumors spread so quickly in social networks, Commun. ACM 55 (6) (2012) 70–75.
[3] M. Kitsak, L.K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H.E. Stanley, H.A. Makse, Identification of influential spreaders in complex networks, Nat. Phys. 6 (11) (2010) 888–893.
[4] S. Pei, H.A. Makse, Spreading dynamics in complex networks, J. Stat. Mech. Theory Exp. 2013 (12) (2013) P12002.
[5] S. Pei, L. Muchnik, J.S. Andrade Jr., Z. Zheng, H.A. Makse, Searching for superspreaders of information in real-world social media, Sci. Rep. 4 (2014).
[6] N.A. Christakis, J.H. Fowler, Social network sensors for early detection of contagious outbreaks, PLoS One 5 (9) (2010) e12948.
[7] Y.H. Fu, C.Y. Huang, C.T. Sun, Using global diversity and local features to identify influential social network spreaders, in: 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2014, pp. 948–953.
[8] Y.H. Fu, C.Y. Huang, C.T. Sun, Using global diversity and local topology features to identify influential network spreaders, Physica A 433 (2015) 344–355.
[9] M. Newman, Networks: An Introduction, OUP Oxford, 2010.
[10] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank citation ranking: bringing order to the web, 1999.
[11] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440–442.
[12] A.L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[13] C. Song, S. Havlin, H.A. Makse, Self-similarity of complex networks, Nature 433 (7024) (2005) 392–395.
[14] L.K. Gallos, C. Song, H.A. Makse, A review of fractality and self-similarity in complex networks, Physica A 386 (2) (2007) 686–691.
[15] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[16] M.E. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci. 103 (23) (2006) 8577–8582.
[17] M.E. Newman, Communities, modules and large-scale structure in networks, Nat. Phys. 8 (1) (2012) 25–31.
[18] M. Rosvall, C.T. Bergstrom, An information-theoretic framework for resolving community structure in complex networks, Proc. Natl. Acad. Sci. 104 (18) (2007) 7327–7331.
[19] T. Teitelbaum, P. Balenzuela, P. Cano, J.M. Buldú, Community structures and role detection in music networks, Chaos 18 (4) (2008) 043105.
[20] X. Zhang, J. Zhu, Q. Wang, H. Zhao, Identifying influential nodes in complex networks with community structure, Knowl.-Based Syst. 42 (2013) 74–84.
[21] R. Shang, S. Luo, Y. Li, L. Jiao, R. Stolkin, Large-scale community detection based on node membership grade and sub-communities integration, Physica A 428 (2015) 279–294.
[22] S.P. Borgatti, Identifying sets of key players in a social network, Comput. Math. Organ. Theory 12 (1) (2006) 21–34.
[23] T.H. Cormen, Introduction to Algorithms, MIT press, 2009.
[24] K. Deb, A. Pratap, S. Agarwal, T.A.M.T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[25] T. He, K.C. Chan, Evolutionary community detection in social networks, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1496–1503.
[26] C.Y. Huang, T.H. Wen, Optimal installation locations for automated external defibrillators in Taipei 7-Eleven stores: using GIS and a genetic algorithm with a new stirring operator, Comput. Math. Methods Med. 2014 (2014).
[27] Y.S. Tsai, P.C.I. Ko, C.Y. Huang, T.H. Wen, Optimizing locations for the installation of automated external defibrillators (AEDs) in urban public streets through the use of spatial and temporal weighting schemes, Appl. Geogr. 35 (1) (2012) 394–404.
[28] M. Tasgin, A. Herdagdelen, H. Bingol, Community detection in complex networks using genetic algorithms, 2007. arXiv preprint arXiv:0711.0491.
[29] S.J. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 2010.
[30] J. Ji, X. Song, C. Liu, X. Zhang, Ant colony clustering with fitness perception and pheromone diffusion for community detection in complex networks, Physica A 392 (15) (2013) 3260–3272.
[31] M. Gong, Q. Cai, X. Chen, L. Ma, Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition, IEEE Trans. Evol. Comput. 18 (1) (2014) 82–97.
[32] F. Morone, H.A. Makse, Influence maximization in complex networks through optimal percolation, Nature (2015).
[33] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110.
[34] A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis, Phys. Rev. E 80 (5) (2009) 056117.
[35] J. Handl, J. Knowles, An evolutionary approach to multiobjective clustering, IEEE Trans. Evol. Comput. 11 (1) (2007) 56–76.
[36] J. Handl, J. Knowles, Multiobjective clustering with automatic determination of the number of clusters. UMIST, Manchester, Tech. Rep. TR-COMPSYSBIO-2004–02, 2004.
[37] N. Matake, T. Hiroyasu, M. Miki, T. Senda, Multiobjective clustering with automatic k-determination for large-scale data, in: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, ACM, 2007, pp. 861–868.
[38] M. Gong, B. Fu, L. Jiao, H. Du, Memetic algorithm for community detection in networks, Phys. Rev. E 84 (5) (2011) 056101.
[39] M. Gong, L. Ma, Q. Zhang, L. Jiao, Community detection in networks by using multiobjective evolutionary algorithm with decomposition, Physica A 391 (15) (2012) 4050–4060.
[40] C. Pizzuti, GA-NET: A genetic algorithm for community detection in social networks, in: Parallel Problem Solving from Nature–PPSN X, Springer Berlin, Heidelberg, 2008, pp. 1081–1090.
[41] C. Pizzuti, A multi-objective genetic algorithm for community detection in networks, in: 21st International Conference on Tools with Artificial Intelligence, 2009, ICTAI'09, IEEE, 2009, pp. 379–386.
[42] C. Pizzuti, A multiobjective genetic algorithm to find communities in complex networks, IEEE Trans. Evol. Comput. 16 (3) (2012) 418–430.
[43] R. Shang, J. Bai, L. Jiao, C. Jin, Community detection based on modularity and an improved genetic algorithm, Physica A 392 (5) (2013) 1215–1231.
[44] R. Shang, S. Luo, W. Zhang, R. Stolkin, L. Jiao, A multiobjective evolutionary algorithm to find community structures based on affinity propagation, Physica A 453 (2016) 203–227.

[45] B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (5814) (2007) 972–976.
[46] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. (1977) 452–473.
[47] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations, Behav. Ecol. Sociobiol. 54 (4) (2003) 396–405.
[48] V. Krebs, unpublished. http://www.orgnet.com/ [accessed: 18.06.16].
[49] L.A. Adamic, N. Glance, The political blogosphere and the 2004 US election: divided they blog, in: Proceedings of the 3rd International Workshop on Link Discovery, ACM, 2005, pp. 36–43.
[50] L. Danon, A. Diaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech. Theory Exp. 2005 (09) (2005) P09008.
[51] C.Y. Huang, C.T. Sun, C.Y. Cheng, Y.S. Tsai, Resource limitations, transmission costs and critical thresholds in scale-free networks, in: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Vol. 2, International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 1121–1128.
[52] R. Pastor-Satorras, A. Vespignani, Epidemic dynamics and endemic states in complex networks, Phys. Rev. E 63 (6) (2001) 066117.
[53] C. Castellano, R. Pastor-Satorras, Thresholds for epidemic spreading in networks, Phys. Rev. Lett. 105 (21) (2010) 218701.