International Journal of Modern Physics C
Vol. 30, No. 6 (2019) 1950050 (21 pages)

DOI: 10.1142/S0129183119500505

An adaptive discrete particle swarm optimization for influence maximization based on network community structure

Jianxin Tang*,†,‡, Ruisheng Zhang†,§, Yabing Yao*, Zhili Zhao†,
Baoqiang Chai† and Huan Li†

*School of Computer and Communication
Lanzhou University of Technology
Lanzhou, Gansu 730050, P. R. China

†School of Information Science and Engineering
Lanzhou University, Lanzhou, Gansu 730000, P. R. China
‡tangjx16@lzu.edu.cn
§zhangrs@lzu.edu.cn

Received 26 November 2018
Accepted 13 May 2019
Published 4 July 2019

As an important research field of social network analysis, influence maximization problem is targeted at selecting a small group of influential nodes such that the spread of influence triggered by the seed nodes will be maximum under a given propagation model. It is yet filled with challenging research topics to develop effective and efficient algorithms for the problem especially in large-scale social networks. In this paper, an adaptive discrete particle swarm optimization (ADPSO) is proposed based on network topology for influence maximization in community networks. According to the framework of ADPSO, community structures are detected by label propagation algorithm in the first stage, then dynamic encoding mechanism for particle individuals and discrete evolutionary rules for the swarm are conceived based on network community structure for the meta-heuristic optimization algorithm to identify the allocated number of influential nodes within different communities. To expand the seed nodes reasonably, a local influence preferential strategy is presented to allocate the number of candidate nodes to each community according to its marginal gain. The experimental results on six social networks demonstrate that the proposed ADPSO can achieve comparable influence spread to CELF in an efficient way.

Keywords: Social networks; viral marketing; influence maximization; community detection; adaptive discrete particle swarm optimization.

PACS Nos.: 89.20.Ff, 89.75.Fb.

1. Introduction

Viral marketing[1] has been showing greater vitality since its meet cute with the powerful social networks, which have become prevalent and important platforms for

‡Corresponding author.

information diffusion in recent years. A growing number of diffusible activities are preferentially to be promoted on social networks by taking advantage of the effect of "word-of-mouth" inherent in viral marketing. Some known examples including catchy news, political propaganda and rumor, etc., tend to recruit enormous stirrers for their outbreaking on the social networks. One of the reasons behind this is that there are hub individuals in the network that can always reshape the attitudes and behaviors of their adherents.[2] Therefore, the promotional profit underlying viral marketing could be expected by the enlistment of influential individuals. Domingos and Richardson[3] modeled the network value of the customers in a social network perspective and formulated the expected profit of a viral marketing as an influence maximization problem.

Influence maximization is targeted at selecting a small group of influential nodes from the network as seed set such that the expected spread of influence triggered by the seed nodes is maximum under a propagation model. Kempe *et al.*[4] proved that the influence maximization is an NP-hard problem under the independent cascade (IC) model and linear threshold (LT) model, meanwhile, they proposed a simple greedy algorithm to select the seed nodes based on the hill-climbing strategy for influence maximization. The original greedy algorithm is guaranteed to find a solution but has to simulate the spread of node influence on the network tens of thousands times to estimate the node importance approximately. Therefore, the greedy-based algorithms relying on simulation strategy are computationally expensive and even lead to poor performances under critical mass transition models in large-scale social networks.[7] Novel influence maximization algorithms[6,8–12] that integrate models for node influence evaluation and effective selection mechanisms have been proposed by researchers in the last decade. To achieve a satisfying performance on time computation and memory cost, most algorithms for influence maximization have to make sacrifices on solution accuracy.

Studies showed that "community structure" is a common feature in many social networks.[13,14] Meanwhile, empirical studies also show that the social individuals' influence mainly spreads within the densely connected communities.[7] In this paper, an efficient meta-heuristic algorithm named adaptive discrete particle swarm optimization is proposed specially for community networks to select influential nodes for influence maximization problem under the IC model. The major advantages and contributions of this paper are summarized as follows:

- Community structures in the social network are firstly detected by leveraging the label propagation algorithm (LPA). All the communities are ranked and weighted by the node size of each community to coordinate with the adaptive strategy on seed nodes selection.
- An adaptive discrete particle swarm optimization (ADPSO) with dynamic encoding mechanism is proposed based on network topology to optimize the expected influence spread of candidate nodes. To expand the seed nodes reasonably to independent communities and achieve the maximal marginal gain, a local

influence preferential (LIP) strategy is presented, and the number of candidate nodes is allocated adaptively to each community according to the expected influence spread within the community.

- The experimental results carried out on six social networks show that the proposed ADPSO can achieve comparable influence spread to CELF in an efficient way.

The rest of this paper is organized as follows: a literature review on related work is given in Sec. 2. Section 3 describes the mathematical formulation of the IC model and the influence maximization problem, then the label propagation algorithm and the original particle swarm optimization are introduced briefly. Section 4 proposes an ADPSO for influence maximization based on community detection. The experimental results and analysis are given in Sec. 5. Section 6 concludes this paper.

2. Related Work

Domingos and Richardson[3] studied the expected profit from customers of viral marketing firstly from a network perspective and modeled influence maximization as a Markov random field. Following the seminal work, Kempe *et al.*[4] proposed a hill-climbing greedy algorithm to identify influential nodes to maximize the expected profit, and theoretically proved that the solution achieved by the greedy algorithm can be approximated to within a factor of $(1 - 1/e - \epsilon)$, where parameter e is the base of the natural logarithm and ϵ is any positive real number, i.e. the algorithm is guaranteed to find a solution which is at least a constant fraction 63% of the optimal solution. However, the simple greedy algorithm suffers from the problem of time consuming and cannot be scalable to large-scale networks. By further exploring the "diminishing returns" property exhibiting in submodularity function, Leskovec *et al.*[15] proposed the CELF algorithm, which can significantly improve the efficiency of the simple greedy algorithm through maintaining an influence estimation priority queue and selecting the most influential node using a *lazy-forward* strategy. Yet the method tends to undergo huge computation once the cascade probability beyond the critical condition or on large-scale networks. Therefore, how to evaluate influential nodes effectively and select a seed set efficiently to maximize the spread of influence still remains an open research topic.

Besides, the conventional centrality based methods, such as high degree centrality,[4,16] betweenness centrality,[16,17] closeness centrality,[18] as well as the PageRank[19] and LeaderRank,[20] that measure a node's influence by considering its local or global topology, advanced heuristic algorithms were proposed in the last decade. By removing the edges not for propagation, Chen *et al.*[5] proposed two greedy-based algorithms based on an improved IC model and the weighted cascade model, respectively. However, experiments showed that the improved greedy algorithms are still computational expensive and may not be suitable for large networks, so they considered that heuristic methods based on network topology may be promising ways to identify influential nodes. Then, the heuristic SingleDiscount and

DegreeDiscount were conceived according to the ideology that the expected influence spread of a targeted seed node should be discounted by a certain degree once its adjacent neighbors have been selected as seed nodes. Cao *et al.*[21] transformed influence maximization into an optimal resource allocation problem, and proposed a dynamic programming algorithm to find the optimal allocation scheme for seed nodes in disconnected communities. Goyal *et al.*[8] proposed the SIMPATH method in which the spread of influential nodes is estimated by enumerating the simple paths starting from the candidate nodes. In addition, a threshold factor was utilized to adjust the direct size of neighborhood to make a trade-off between efficiency and solution accuracy. Mo *et al.*[9] proposed an existing evidential centrality (EVC) by combining the node degree centrality and edge weight strength based on the DempsterShafer evidence theory to select influential nodes in weighted networks. Bian and Deng[22] improved the EVC metric by integrating the global structure information of the node in the network and proposed a new evidential centrality (NEC) for influential nodes identification. Following the study that the global influence of a node in the network depends on its influence spread within the two-hop area,[23] Gong *et al.*[10] formulated a local influence estimation LIE measurement to approximate the expected influence spread of a node and proposed the DPSO algorithm to optimize the LIE fitness value. Experiments showed that the LIE metric performs better than the expected diffusion value (EDV) metric,[24] which only considers the one-hop neighbors of influential nodes. By taking the advantages of metaheuristic optimization algorithms that evolve based on swarm intelligence, Tang *et al.*[25] proposed an efficient discrete bat algorithm specially for influence maximization problem. To relieve the time computation in networks with massive size of nodes, Borgs *et al.*[26] proposed a reverse influence sampling (RIS) method based on random sampling theory to identify the k most influential nodes. Theoretical analysis proved that the algorithm can obtain a near-optimal approximation factor of $(1 - 1/e - \epsilon)$ in nearly optimal time. More recently, two novel sampling frameworks named SSA and D-SSA were proposed by Nguyen *et al.*,[27] which are up to 1200 times faster than the IMM method[28] while providing the same approximation guarantee according to the experimental results. However, experiments showed that SSA tends to achieve suboptimal solutions with the increase of the targeted seed set size.[25]

As stated by Newman *et al.*[13] community structure is an ubiquitous feature shared by many social networks. Nodes within a community connect and interact with others densely while the connections between communities are sparser. Studies[7,21,29] demonstrated that community structure plays an important role in information diffusion, and the influence mainly spreads within densely connected communities. Wang *et al.*[30] proposed a community-based greedy algorithm (CGA) to find the top-k influential nodes, in which the dynamic programming mechanism was adopted to select targeted communities, then the greedy strategy was employed for the selection of seed nodes within limited communities. Nevertheless, empirical experiments show that CGA is computationally expensive in networks with large-scale communities. Kundu and Pal[31] proposed a deprecation-based greedy strategy

(DGS) to select influential nodes in community networks. The influence of each candidate node is approximated by an integrated centrality metric in its appurtenant community, and nodes with lower influence spread are marked to be deprecated. Rahimkhani *et al.*[32] proposed the *ComPath* algorithm for influence maximization. According to the framework of *ComPath*, communities in the original network were treated as individual nodes to construct a new graph, through which the most central nodes, i.e. crucial communities, were identified based on betweenness centrality measure. Then, the top-k nodes were selected among the crucial communities based on the path metric. Shang *et al.*[33] adopted the simple greedy strategy to approximate the expected influence of a node within its own community and proposed a community-based framework for influence maximization (CoFIM). To improve the efficiency of the greedy strategy in large-scale networks, Shang *et al.*[34] promoted the IMPC algorithm by further expanding the seed nodes and derived a novel influence estimator based on multi-neighbor potential of node in community networks.

Developing efficient algorithms for influence maximization problem without loss of solution quality is still filled with challenging research topics especially in large-scale networks. In this paper, a meta-heuristic algorithm named adaptive discrete particle swarm optimization is proposed for influence maximization problem based on community detection. According to the framework, community structures of the input network are detected firstly using the fast LPA. Dynamic encoding mechanism for particle individual and discrete evolutionary rules for the virtual particle swarm are conceived according to network community structure, respectively. Meanwhile, an LIP strategy is presented to adjust the particle encoding scheme adaptively according to the expected influence spread value within each community. Then, the LIE estimator, which is adopted to evaluate the expected influence spread of candidate seed set, is optimized by the ADPSO to find the global optimal seed set for influence maximization.

3. Preliminaries

3.1. *Propagation cascade model*

IC model[4] is a probability model which mimics the spread process of information in social networks. According to IC model, each node in the network has only two states, either *active* or *inactive*. Nodes can be allowed to switch from *inactive* to *active* ones, but not *vice versa*. Propagation probability p describes the tendency of inactive individuals to be affected by its adjacent active neighbors.

Given an active node set S, at step t, for an active node u from S, it has only one chance to activate each of its adjacent inactive neighbors v and successes with a probability p_{uv}, which is associated with edge $(u, v) \in E$. Whether the activation is succeed or not, node u will no more attempt to activate v in the following steps. If node v is activated by u, then v will remain active and has one chance to activate each of its adjacent inactive neighbors in step $t + 1$. The diffusion process terminates

if no node is activated at step T and returns the influence spread $\sigma(S)$ comprising all of the active nodes.

3.2. *Influence maximization*

Definition 1. Let $G = (V, E, W)$ be a network, V is the set of nodes and E is the set of edges in the network. Weight set W maps each edge $(u, v) \in E$ to its influence spread probability $w_{u,v} \in (0, 1)$, which can be predefined according to the given influence propagation model or learned from spreading rules problem-specially.

Definition 2. Given a network $G = (V, E, W)$ and a quota on the size of K $(1 \leq K < |V|)$, influence maximization problem aims to select K influential nodes into seed set S such that the expected number of influenced nodes triggered by the seed set, denoted as influence spread $\sigma(S)$, is maximum under a given propagation cascade model.

$$S^* = \underset{S \subseteq V, |S|=K}{\arg\max}\, \sigma(S). \tag{1}$$

In Eq. (1), S is a candidate seed set, $\sigma(S)$ is the expected number of influenced nodes that are triggered by S, and S^* is the best seed set that can maximize the spread of influence.

3.3. *Label propagation algorithm*

In this paper, the simple and fast LPA investigated by Raghavan *et al.*[35] is employed to detect communities of a given network. Unlike the traditional community detecting methods that require *priori information* or a pre-defined objective optimization function, LPA only needs the network structure as its guide.

The ideology of LPA can be described as follows: initially, each node in the network is given an unique label. At each iteration, a sweep over all nodes is performed in random sequential order, during which each node takes the label shared by the majority of its direct neighbors. If there is no unique majority, one of the majority labels is picked randomly. As the labels propagate across the network, majority labels will replace most labels and dominate the network. The process will be terminated when each node is traced by the majority label of its neighbors, then independent communities are defined as groups of nodes with identical labels at convergence.

3.4. *Influence spread estimator*

Zhu *et al.*[6] pointed out that evaluating the expected influence spread of a given node set accurately is a ♯P-hard problem. As mentioned in Sec. 2, a local influence estimator (LIE) function based on the Two-Degree theory[23] was formulated to approximate the expected influence spread of given node set. The mathematical formulation of LIE id described in Eq. (2), where $\sigma_0(S)$ is the K most influential

nodes in the seed set S, $\sigma_1^*(S)$ and $\tilde{\sigma}_2(S)$ are the expected influence spread of one-hop and two-hop areas of the seed set, respectively. Correspondingly, $N_S^{(1)}$ and $N_S^{(2)}$ represent the one-hop and two-hop areas of seed set S. p_u^* is a small constant active probability of different diffusion models. d_u^* is the number of edges of node u within $N_S^{(1)}$ and $N_S^{(2)}$. Therefore, the goal of influence maximization problem can be settled by selecting a quota number of influential nodes to maximize the fitness value of LIE function.

$$\begin{aligned} \text{LIE} &= \sigma_0(S) + \sigma_1^*(S) + \tilde{\sigma}_2(S) \\ &= K + \sigma_1^*(S) + \frac{\sigma_1^*(S)}{|N_S^{(1)} \setminus S|} \sum_{u \in N_S^{(2)} \setminus S} p_u^* d_u^* \\ &= K + \left(1 + \frac{1}{|N_S^{(1)} \setminus S|} \sum_{u \in N_S^{(2)} \setminus S} p_u^* d_u^*\right) \sum_{i \in N_S^{(1)} \setminus S} \left(1 - \prod_{(i,j) \in E, , \in S} (1 - p_{i,j})\right). \end{aligned} \tag{2}$$

3.5. *Particle swarm optimization*

Particle swarm optimization (PSO), which was proposed by Kennedy and Eberhart,[36] is a bio-inspired meta-heuristic optimization algorithm mimicking the foraging behavior of bird flocks in the wild. By idealizing the behavior of bird flocks, PSO can be illustrated as a population of N particles with respective position and velocity vectors fly to forge the food that locates in a d-dimensional searching space of the problem to be solved. During the searching process, particles can learn from each other to update their position and adjust the "flying" directions until they finally converge to the global optimization. The original mathematical model of PSO is formulated as follows:

$$V_i^{t+1} = V_i^t + c_1 r_1 (Pb_i - X_i^t) + c_2 r_2 (Gb - X_i^t), \tag{3}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}, \tag{4}$$

where $V_i = (v_{i1}, v_{i2}, \ldots, v_{id})$ and $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ are position and velocity vectors of particle i $(i = 1, 2, \ldots, N)$ at time t, respectively. c_1 and c_2 are two constant learning factors. r_1 and r_2 are two random coefficients drawn uniformly from $(0, 1)$. Pb_i denotes the previous historical best position of particle i, and Gb denotes the global best position of the swarm up to the current generation. The second and the third section at the righthand side of Eq. (3) are known as "self-learning" and "social cognition" factors, respectively, to adjust the trajectories of the swarm to global optimal solution.

The effectiveness and robustness of WPSO has been widely validated by tackling many continuous and combinational optimization problems.[37–40] Among the state-of-the-art works on PSO, Shi *et al.*[38] proposed a notable weighted particle swarm optimization (WPSO), of which the mathematical model of the velocity vector is formulated as in Eq. (5), where inertia weight ω is adopted to regulate the effect of

the historical best velocity of particle i on its current velocity updating. It means that a larger inertia weight ω facilitates global exploration while a smaller inertia weight ω tends to facilitate local exploitation to fine-tune the current searching area.

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 (Pb_i - X_i^t) + c_2 r_2 (Gb - X_i^t). \tag{5}$$

4. Proposed Method

4.1. *Framework of ADPSO for influence maximization*

As discussed aforementioned, the local influence estimator described in Eq. (2) is a discrete optimization problem which is designed to evaluate the expected influence spread of a given seed set. So, the WPSO algorithm can be employed and modified according to the discrete network searching space to find the K most influential nodes for influence maximization. Figure 1 shows the framework of the proposed ADPSO for influence maximization. There are two main operations including community detection and candidate nodes refinement in the framework.

A detailed description about the operations in the framework is given in the following seven steps:

Step 1. Community detection. The community structure set of the input network G is firstly detected using LPA algorithm, denoted as $C = \{C_1, C_2, \ldots, C_M\}$, where M is the number of communities in the set.

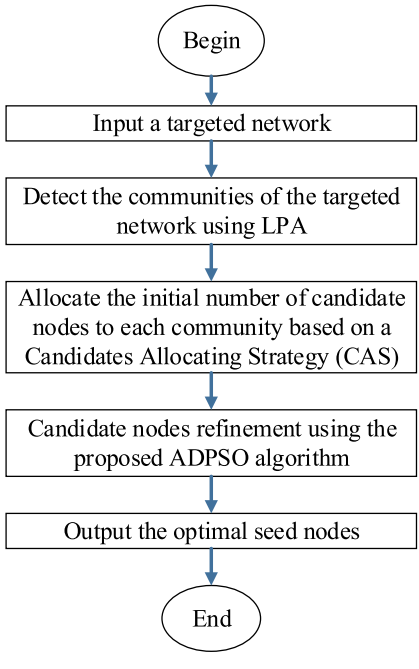


Fig. 1. (Color online) The framework of the proposed ADPSO for influence maximization problem.

Step 2. Candidate nodes allocation. Initially, to select the targeted K most influential nodes from the M communities for influence maximization, an intuitive idea is to allocate a computed number of candidate nodes to each community in proportion to its node size, as

$$k_l(C_l) = \frac{|C_l|}{\sum_{l=1}^{M} |C_l|} * K, \tag{6}$$

where k_l is a functional factor and returns the number of allocated candidates for community C_l $(l = 1, 2, \ldots, M)$, $|C_l|$ is the node size of community C_l. Meanwhile, the sum of k_l meets the constraint condition

$$\sum_{l=1}^{M} k_l = K. \tag{7}$$

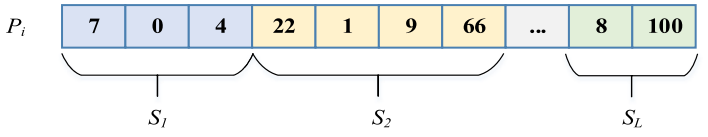
However, a potential problem faced by this allocation scheme is that k_l tends to be 0 and the sum of k_l will be less than K in large-scale networks, in which there are always a large number of homogeneous communities such that the node size of each community accounts for merely a tiny proportion of the total number of nodes in the network. Therefore, a novel candidates allocating strategy (CAS) is presented in this paper to allocate the initial number of candidate nodes to each community in a more reasonable way by considering the relationship between K and M. First, the M communities are ranked in decreasing order according to the node size of each community. Then, the number of candidate nodes for each community is allocated according to the following two scenarios:

(i) If $M > K$, then the first K communities are selected from the ordered sequence and the number of candidate nodes allocated to each of the K communities is set to 1, respectively.
(ii) Otherwise, i.e. $M \leq K$, the number of candidate nodes is allocated to each community based on a preferential allocating method, shown in Eq. (8). More specifically, the number of candidate nodes allocated to each of the M communities is r, and if $K - r * M < M$, then allocate one more candidate node to the first $(K - r * M)$ communities preferentially.

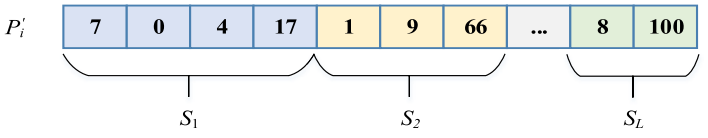
$$k_l(C_l) = \text{Prior}(K - r * M) + r, \quad r = \left\lfloor \frac{K}{M} \right\rfloor. \tag{8}$$

In view of the above two scenarios, let the upper bound of l be L, where $k_l(C_l) > 0$ $(l = 1, 2, \ldots, L)$. We can see that the presented CAS strategy can ensure that the sum of k_l meets the constraint condition in Eq. (7) after the allocation operation is finished.

Step 3. Particle swarm initialization. To tackle the influence maximization in community networks using the meta-heuristic PSO, discrete encoding scheme needs to be constructed based on the network topology structure. According to the CAS strategy presented above, $k_l(C_l)$ candidate nodes are firstly drawn randomly from



(a) An illustration of the encoding scheme for the initialization of particle P_i



(b) An illustration of the recoding scheme for particle P_i

Fig. 2. (Color online) Illustration of the encoding scheme for particle P_i during the evolutionary iterations.

community C_l to initialize the lth subset S_l of the particle P_i $(i = 1, 2, \ldots, N)$ in the swarm. Then, the expected influence spread of k_l candidate nodes from community C_l in particle P_i is estimated separately, meanwhile, record the initial k_l candidate nodes with the maximal expected influence spread of community C_l as Pb_l and calculate the whole expected influence spread of the best L $(L \leq M)$ communities as Gb.

A toy example of the encoding scheme for the initialization of particle P_i is illustrated in Fig. 2(a). According to the illustration, $S_1 = \{7, 0, 4\}$ is the corresponding candidate node set of the first community C_1, and all the subsets S_l make up an objective seed set $S = \{S_1, S_2, \ldots, S_L\}$. The number of boxes with the same color equals to $k_l(C_l)$, i.e. the number of candidate nodes allocated to community C_l according to the CAS strategy, the integer digit in each box is the unique identification of the corresponding node in the network.

Step 4. Global exploration. Apparently, the initial particles need to be improved by exploring more influential nodes in the separately community space. For this purpose, a LIP strategy is presented in this paper to expand the number of seed nodes reasonably and adaptively according to the marginal gain of the subsets from different communities: for every community C_l from each particle P_i, an additional potential candidate node with the highest degree centrality but not in the current candidate node set S_l is selected and added to S_l temporarily. Then, the new expected influence spread of the $k_l(C_l) + 1$ candidate nodes in S_l is estimated according to the LIE estimator, and the average marginal gain of the new added temporal node is calculated. Consequently, the node that brings the maximal marginal gain is preserved in its subordinative candidate set. Meanwhile, the candidate node with the lowest degree centrality is to be removed accordingly from its subordinative set where the average marginal gain is the least.

Step 5. Particle swarm recoding. Once the two candidate node sets that need add an extra candidate node and remove a less influential candidate node are verified, respectively, in *Step 4*, the encoding constitution corresponding to the two candidate

sets of particle P_i needs to be adjusted by adding the new selected candidate node with maximal marginal gain into its subordinative candidate node set and by removing the node with the lowest node degree centrality from its corresponding candidate set, respectively. Then, Pb_i and Gb are updated after the recoding operations of the N particles are finished.

The toy example illustrated in Fig. 2(a) is recoded based on the dynamic recoding scheme and illustrated in Fig. 2(b). At generation g, the new temporal node 17 is verified to be added into the candidate set S_1 of the first community of particle P_i, and node 22 is verified to be removed from the candidate set S_2 of the second community, therefore, the encoding constitution corresponding to the first community is recoded by adding the node 17, $S_1 = \{7, 0, 4, 17\}$. At the same time, the encoding constitution corresponding to the second community is recoded by removing the node 22, $S_2 = \{1, 9, 66\}$. And other candidate sets remain unchanged.

Step 6. Local improvement. To improve the current global best virtual particle, a refinement on the L candidate sets from Gb is carried out by ADPSO separately.

Step 7. Check convergence. The algorithm will be terminated if it has run the predefined maximal generations. Otherwise, go to *Step* 4.

4.2. *Adaptive discrete particle swarm optimization*

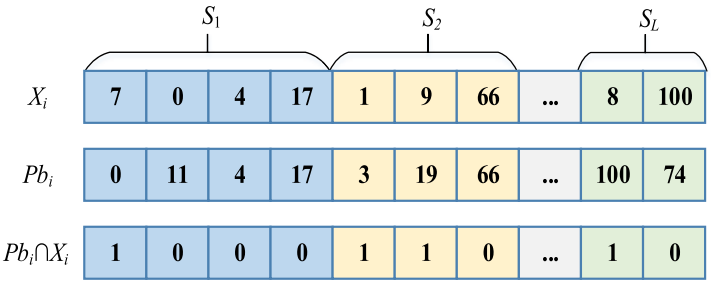
According to the particle swarm initialization strategy, as described in *Step* 3 in Sec. 4.1, discrete encoding scheme and dynamic recoding scheme are conceived for particle individuals, as shown in Fig. 2. Therefore, novel evolutionary rules for velocity and position vectors are also needed to be constructed in discrete forms based on network topology characteristic to coordinate with the dynamic encoding scheme adaptively, as

$$V_i^{t+1} \leftarrow H(\omega V_i^t + c_1 r_1 (Pb_i \cap X_i^t) + c_2 r_2 (Gb \cap X_i^t)), \tag{9}$$

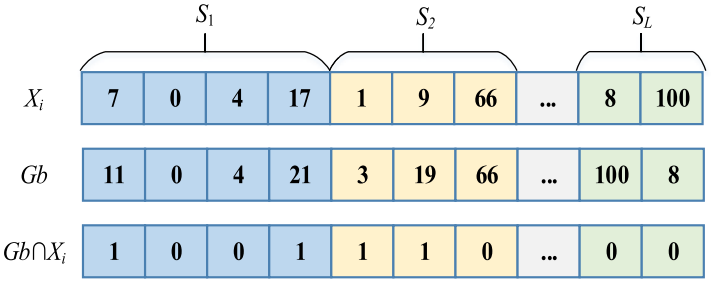
$$X_i^{t+1} \leftarrow X_i^t \oplus V_i^{t+1}. \tag{10}$$

Under the discrete evolutionary rules, position vector X_i^t is encoded by K candidate nodes from the first L communities, and so do the vectors Pb_i and Gb. Operator "$\cap$" is defined as a logical similar intersection operation, through which the intersecting result is a vector composed by 0 and 1, where 0 represents the element S_l^j $(l = 1, 2, \ldots, L,\ j = 1, 2, \ldots, k_l)$ in the lth candidate node set S_l from X_i has an identical element in the set S_l from Pb_i or Gb, 1 is the otherwise. Velocity vector V_i^t is encoded by 0 and 1, where 0 represents the node corresponding to the position at S_l from X_i is a targeted influential node and can be preserved for the evolution of the next generation, 1 represents the node corresponding to the position at S_l from X_i needs to be replaced or to be adjusted by another potential node from community C_l.

$H(\cdot)$ is a velocity control function to ensure that $V_i(l, j)$ is 0 or 1. Assuming that the parameter is X_i, $H(X_i)$ can be represented as $H(X_i) = (h(S_1^1), \ldots, h(S_1^{k_1}), h(S_2^1), \ldots, h(S_2^{k_2}), \ldots, h(S_L^{k_L}))$, where $h(S_l^j)$ is defined as a threshold



(a) Illustration of the "∩" operation on Pb_i and X_i



(b) Illustration of the "∩" operation on Gb and X_i

Fig. 3. (Color online) Illustration of the "∩" operation on X_i with Pb_i and Gb.

factor formulated as

$$h(S_l^j) = \begin{cases} 0 & \text{if } S_l^j < 2, \\ 1 & \text{if } S_l^j \geq 2. \end{cases} \tag{11}$$

An example given in Fig. 3 shows the "⊕" operation on X_i with Pb_i and Gb. Assumed that there are three communities in the network, if we set $K = 9$ and fix the coefficients $c_1 r_1 = 0.6$ and $c_2 r_2 = 1.5$, then the velocity vector V_i can be calculated as $V_i = H(0.6 * (1,0,0,0,1,1,0,1,0) + 1.5 * (1,0,0,1,1,1,0,0,0)) = H((2.1,0,0,1.5, 2.1,2.1,0,0.6,0)) = (1,0,0,0,1,1,0,0,0)$.

According to Eq. (10), the operator "⊕" is adopted to judge whether the element in X_i should be kept or adjusted based on the value of V_i, as

$$X_i'(l,j) = \begin{cases} X_i(l,j) & \text{if } V_i(l,j) = 0, \\ \text{Replace}(X_i(l,j), C_l) & \text{if } V_i(l,j) = 1, \end{cases} \tag{12}$$

where $X_i(l,j)$ is the jth element of the candidate set S_l in X_i. Replace$(\cdot)$ is a function that replaces the element $X_i(l,j)$ with a random node drawn from community C_l and guarantees there is no repeated node in S_l from X_i after the replacement is finished.

Based on the dynamic encoding scheme and discrete evolutionary rules for the particle swarm, the adaptive particle swarm optimization for the influence

Algorithm 1. ADPSO for influence maximization based on community detection.

Input: Network $G = (V, E)$, maximum number of generations $g_{\max}$, particle swarm size N, inertia weight ω, learning factors c_1 and c_2.

Detect the communities of the given network using LPA
Draw k_l candidate nodes randomly from community C_l separately to initialize position vectors X and update Pb
Initialize velocity vector $V \leftarrow 0$
Select out the initial global best position vector into Gb according to the LIE value of each community C_l, and the best expected influence spread $Gbest$
Initialize iterator $g \leftarrow 0$
while $g < g_{\max}$ **do**
    Add one more candidate to community C_l of particle P_i and calculate the
    marginal gain
    Recode the particle swarm, Pb and Gb according the marginal gain
    Update the velocity vector V according to Eq. (9)
    Update the position vector X according to Eq. (10)
    Update the Pb and select out the global best Gb of the current generation,
    update $Gbest$
    Employ the local search operation, $Gb^{'} \leftarrow LocalSearch(Gb)$
    Calculate the new $Gbest^{'}$ and update the $Gbest, Gbest \leftarrow \max(Gbest, Gbest^{'})$
    $g \leftarrow g + 1$
end while
return Gb as the optimal seed set S

maximization problem in community networks is given in Algorithm 1. During the evolutionary process, a local improvement strategy, as shown in Algorithm 2, is adopted to improve the candidate seed set Gb of the current generation. After the algorithm runs $g_{\max}$ generations, Gb is decoded as the global optimal seed set S^* for influence maximization.

In Algorithm 2, function $DC(C_l)$ returns the degree centrality vector of each node in the community C_l from current global best virtual particle Gb. Function $Order(C_l^*, d_{C_l})$ returns an ordered node set on the basis of the degree centrality of each node in ascending order. Function $Length(Nei\_set)$ returns the size of the neighbor set Nei_set. $Replace(node_i, Nei\_set)$ is adopted to replace $node_i$ with one of its one-hop area neighbors and guarantees that there is no repeated nodes after the replacement is finished.

4.3. *Time complexity of the method*

The time complexity of the proposed adaptive particle swarm optimization based on community structure mainly relies on the community detection and the adaptive evolution parts. For the first part, it needs $O(m + n)$ basic operations to detect the

Algorithm 2. *LocalSearch*(Gb)

Input: Gb of the current generation.

```
for C_l ∈ Gb do
    d_{C_l} ← DC(C_l)
    C_l' ← Order(C_l, d_{C_l})
    for node_i ∈ C_l' do
        Nei_set ← N^(1)_{node_i}
        index ← 1
        len ← Length(Nei_set)
        while index ≤ len do
            C_l' ← Replace(node_i, Nei_set)
            if LIE(C_l') > LIE(C_l) then
                C_l ← C_l'
            end if
            index ← index + 1
        end while
    end for
end for
return the newly global best seed set Gb
```

communities of a given network using the standard LPA, where m is the number of edges and n is the number of nodes in the network. There are several complex steps in the adaptive particle swarm optimization. First, ranking the M communities needs at least $O(M \log M)$ basic operations. Under the worst case, it needs $O(NM\bar{k})$ to k_l candidate nodes and initialize the particle position vectors from L valid communities, where $\bar{k}$ is the average value of the k_l ($l = 1, 2, \ldots, L$). Meanwhile, initializing the velocity vector requires $O(NK)$ basic operations, and evaluating the fitness value of LIE requires $O(NK\bar{D})$ under the worst condition, where $\bar{D}$ is the average degree value of the node set V. During the iteration process, calculating the marginal gain of the added potential node of the N particles needs $O(NM\bar{k}\bar{D})$, recoding the N particles needs $O(N\bar{k})$ and updating the position and velocity vectors needs $O(NK^2)$. In the local improvement, the worst case to update the Gb requires $O(\bar{k}^2\bar{D}^2)$. Since $\bar{k} < K$, so the worst time complexity of the proposed algorithm is $O(K\bar{D}(NM + K\bar{D})g_{\max} + (m + n))$.

5. Experiments and Analysis

To validate the performance of ADPSO on influence maximization problem, experiments are carried out on six real-world social networks under the IC model at the propagation probability $p = 0.01$.

Table 1. Statistical characteristic of the six social networks. $|V|$ and $|E|$ represent the number of nodes and edges, respectively. $\langle k \rangle$ is the average degree, $\bar{d}$ is the average shortest path distance, $\bar{C}$ represents the average clustering coefficient, AC represents the assortativity coefficient, Q is the modularity of the network, and M is the number of communities.

Networks	$\|V\|$	$\|E\|$	$\langle k \rangle$	$\bar{d}$	$\bar{C}$	AC	Q	M
Email[41]	1133	5451	9.622	3.606	0.254	0.078	0.395	7
Blogs[42]	3982	6803	3.417	6.252	0.493	−0.133	0.798	281
HepTh[43]	9877	25998	23.409	5.264	0.600	0.268	0.619	908
PGP[42]	10680	24316	4.554	7.486	0.440	0.238	0.804	969
AstroPh[44]	18772	198110	21.107	4.194	0.677	0.205	0.298	631
CondMat[44]	23133	186936	16.162	5.352	0.055	0.135	0.639	2071

5.1. *Datasets*

Table 1 shows the six real-world social networks with statistically community structure. Besides the statistical characteristic, the node degree distribution of the six networks is given in Fig. 4.

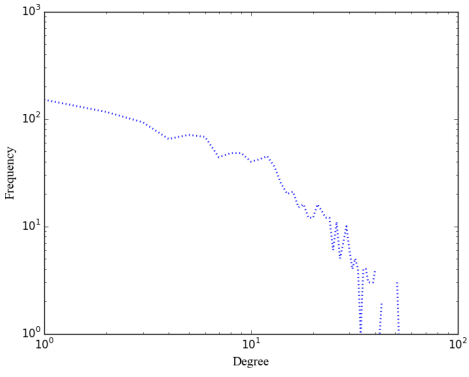
5.2. *Baseline algorithms*

Once the Gb is returned as the global optimal seed set S^* for influence maximization, the corresponding influence spread of S^* is simulated on the six social networks. Meanwhile, four other state-of-the-art algorithms are chosen to compare with ADPSO to validate the performance of the proposed algorithm.

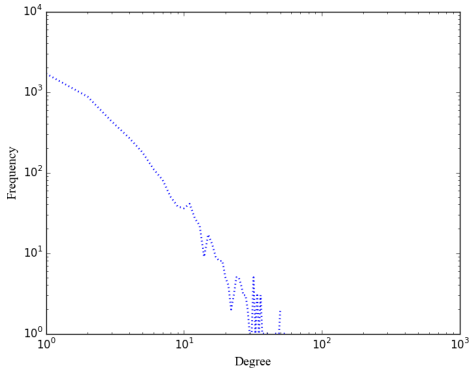
- **CELF**[15] is an improved greedy algorithm with a "lazy-forward" strategy by further exploiting the submodularity property of the influence estimation function. Though CELF is effective in identifying influential nodes for influence maximization, it turns out to be inefficient in large-scale networks.
- **IMPC**[34] is an influence maximization framework based on multi-neighbor potential of nodes in community networks.
- **DDSE**[11] is a discrete meta-heuristic optimization algorithm originated from differential evolutionary ideology. The algorithm evaluates the expected influence spread according the EDV estimator and selects influential nodes based on a degree-descending search strategy for influence maximization.
- **SSA**[27] is a RIS-based method, and the algorithm stops at exponential check points to verify if there is adequate statistical evidence on the solution.

5.3. *Influence spread comparison*

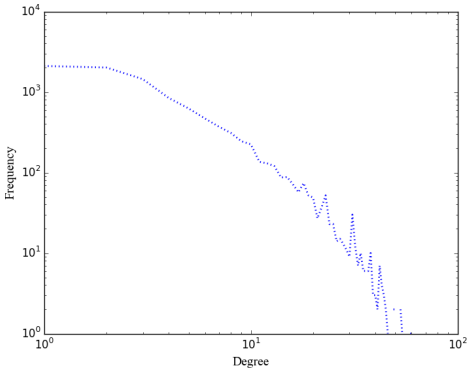
It is important to note that all related parameters values are set according to the suggestions in the original literature when we implement the procedures of the four baseline algorithms. We run the Monte-Carlo simulation 10 000 times for CELF to estimate the marginal gain of each candidate node. The maximal evolutionary generation $g_{\max}$ for ADPSO and DDSE is set as 100, respectively, and the simulation



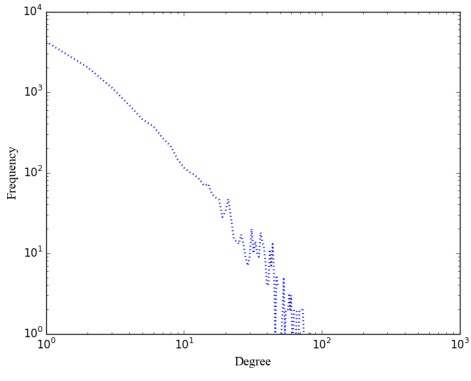
(a) Email



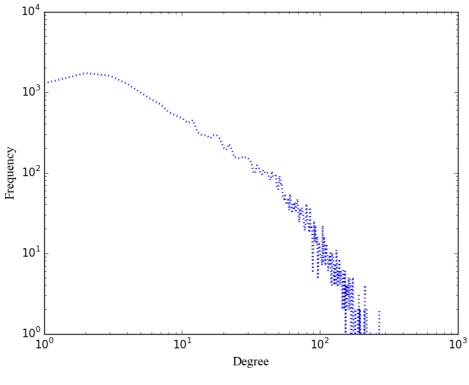
(b) Blogs



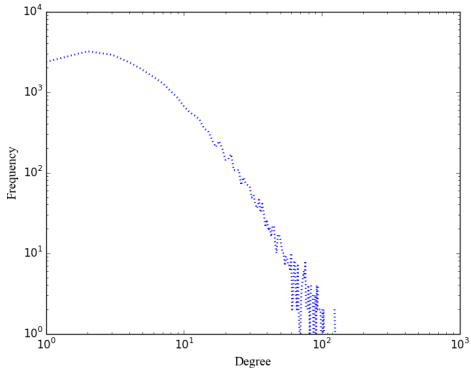
(c) HepTh



(d) PGP



(e) AstroPh



(f) CondMat

Fig. 4. (Color online) Node degree distribution of the six real-world social networks.
Notes: As mentioned above, we gave the Refs. of the six networks. Meanwhile, the information of the six networks can also be found in http://snap.stanford.edu/data/index.html

times for both of the algorithms is set as 1000 to obtain the average influence spread on the targeted network. We set the learning factors $c_1 = c_2 = 2$, and set the inertia weight $\omega = 0.8$ in DPSO. The probabilities of mutation, crossover and diversity operations in DDSE are set as 0.1, 0.4 and 0.6, respectively. For the SSA algorithm, parameters ϵ and δ are set as 0.1 and 0.01, respectively.

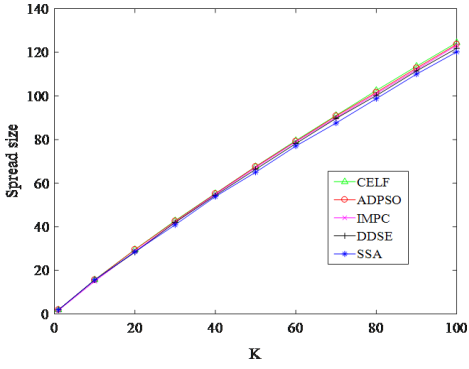
Figure 5 shows the performance comparison on influence spread simulated in the six social networks between ADPSO and other baseline algorithms. From the line charts shown in Fig. 5, we can see that the performance on influence spread of the five algorithms is notably different on the six social networks. Among the five algorithms, the greedy-based CELF performs as the best one for influence maximization except on network Blogs, as shown in Fig. 5(b). Meanwhile, the proposed ADPSO achieves comparable results to CELF and better than IMPC on the six social networks. Inversely, the SSA that relies on reverse influence spreading ideology achieves almost the least influence spread in the targeted networks.

More specially, in the first three social networks with less scale of node size, Figs. 5(a)–5(c) demonstrate that all the five algorithms achieve very closely influence spread, i.e. they can identify the most K influential nodes easily as seed nodes for influence maximization problem. Even in Fig. 5(b), DDSE and ADPSO achieve better solution accuracy than CELF, which further implies that the greedy-based method has no distinct advantages over small networks. However, CELF shows up the guaranteed solution on other three large networks, as shown in Figs. 5(d)–5(f), in which the performance of CELF remains as the best compared with other four algorithms. The interesting reason is that the Monte-Carlo simulation scheme can provide approximately accuracy of the marginal gain, which is insensitive to the scale of targeted networks. However, the node influence spreading estimators used in other four algorithms cannot provide approximately expected influence spread in large-scale networks, so the algorithms tend to achieve less solution than CELF.

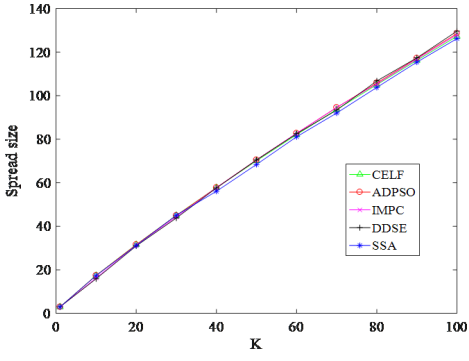
In addition, the influence spread curves of ADPSO grow up more steadily compared with other two algorithms DDSE and SSA with the increase of the seed size K. SSA identifies the most influential nodes effectively at scenarios when the targeted seed size is small, and achieves comparable influence spread to other four algorithms, but tends to return suboptimal seed set when the targeted seed set size is large, as illustrated in Fig. 5(d), where SSA performs less than DDSE when $K > 50$. In Figs. 5(e) and 5(f), the influence spread curves of DDSE are choppy at different scenarios due to its local influence estimator which is overstretched on large-scale networks within one-hop area.

5.4. *Running time comparison*

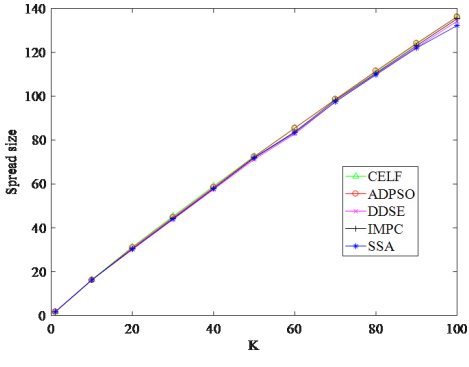
Efficiency is a key factor in developing influence maximization algorithms especially for large-scale networks. To show the efficiency of ADPSO in identifying influential nodes for influence maximization, the running time of the five algorithms at targeted seed set size $K = 100$ on the six networks is given in Fig. 6.



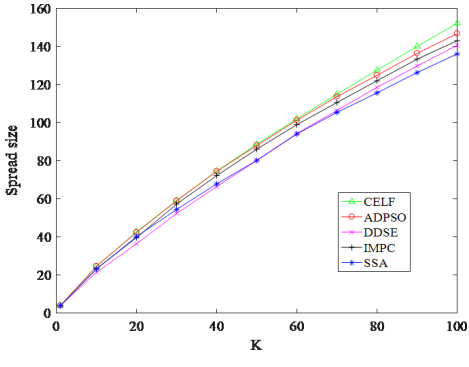
(a) Email



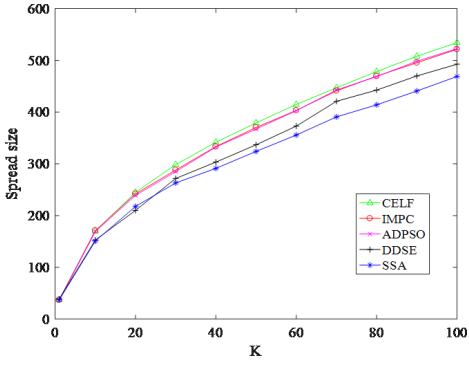
(b) Blogs



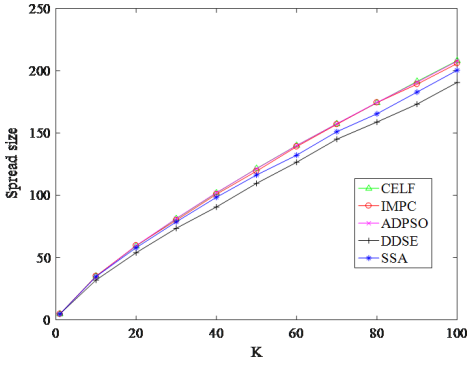
(c) HepTh



(d) PGP



(e) AstroPh



(f) CondMat

Fig. 5. (Color online) Influence spread of the five algorithms under IC model at propagation probability $p = 0.01$ on the six social networks.

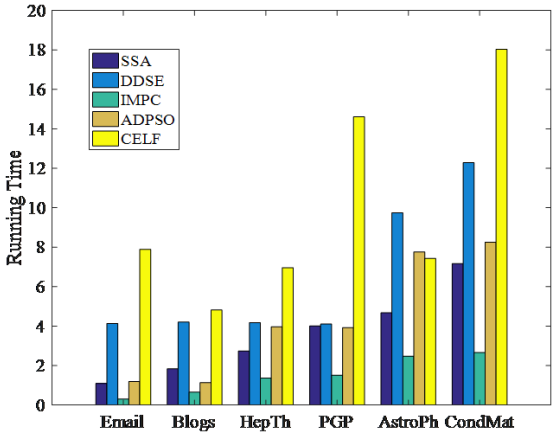


Fig. 6. (Color online) Running time of the five algorithms when to select $K = 100$ seed nodes in the six networks (The running time is measured in seconds, except the CELF, of which the running time is measured in hours on network AstroPh and in minutes on other five networks.).

The bar charts shown in Fig. 6 illustrate that the proposed ADPSO is an efficient algorithm which needs only a few seconds to identify the targeted $K = 100$ seed nodes, and even in the largest social network CondMat, it needs no more than 10 s. By contrast, the greedy-based CELF turns out to be inefficient, for it even needs about 7.4 h to select the targeted $K = 100$ seed nodes in the network AstroPh. The inherent time-consuming feature makes it hard to deal with the influence maximization problem in large-scale networks.

The running time of SSA proves that the algorithm can provide satisfying identification of influential nodes for influence maximization, but it tends to be less effective than CELF, ADPSO and IMPC in consideration of the influence spread shown in Fig. 5. The bar charts also show that DDSE is an efficient algorithm for influence maximization problem. Comparing with ADPSO, the evolutionary rules of DDSE are more complex, but it always returns suboptimal seed set due to the influence estimator used in the algorithm cannot provide approximately influence evaluation within one-hop area of the targeted node.

6. Conclusions

Developing efficient algorithms for influence maximization problem without loss of solution accuracy is still filled with challenging research topics especially in large-scale networks. In this paper, an ADPSO is proposed specially for influence maximization in community networks. The dynamic LIP strategy conceived based on community feature is effective to enable the proposed algorithm to ad-locate reasonable number of candidate nodes adaptively according to the expected marginal gain in the communities with different scale of node size. The experimental results on influence spread under the IC model prove that the network topology based

evolutionary rules and local improvement are efficient in identifying influential seed nodes for influence maximization. Meanwhile, the comparisons on the six real-world networks show that ADPSO is a promising meta-heuristic algorithm for influence maximization.

Acknowledgments

This work is supported by the National Natural Science Foundations of China (Grant Nos. 21503101 and 61702240), the Project-sponsored by SRF for ROCS, SEM (Grant No. SEM[2015]311) and the Fundamental Research Funds for the Central Universities (Project No. lzujbky-2017-191).

References

1. M. Subramani and B. Rajagopalan, *Communications of the ACM — Mobile Computing Opportunities and Challenges* **46**, 300 (2003).
2. J. J. Brown and P. H. Reingen, *J. Consumer Res.* **14**, 350 (1987).
3. P. Domingos and M. Richardson, Mining the network value of customers, in *Proc. Seventh ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, KDD'01* (ACM, New York, NY, USA, 2001), pp. 57–66.
4. D. Kempe, J. Kleinberg and Tardos, Maximizing the spread of influence through a social network, in *ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining* (Washington, DC, USA, 2003), pp. 137–146.
5. W. Chen, Y. Wang and S. Yang, Efficient influence maximization in social networks, in *Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, KDD'09* (ACM, New York, NY, USA, 2009), pp. 199–208.
6. Z. Tian, W. Bai, W. Bin and Z. Chuanxi, *Inf. Sci.* **278**, 535 (2014).
7. A. Galstyan, V. Musoyan and P. Cohen, *Phys. Rev. E* **79**, 056102 (2009).
8. A. Goyal, W. Lu and L. V. S. Lakshmanan, Simpath: An efficient algorithm for influence maximization under the linear threshold model, in *2011 IEEE 11th Int. Conf. Data Mining* (Vancouver, BC, Canada, 2011), pp. 211–220.
9. H. Mo, C. Gao and Y. Deng, *J. Syst. Eng. Electron.* **26**, 381 (2015).
10. M. Gong, J. Yan, B. Shen, L. Ma and Q. Cai, *Inf. Sci.* **367–368**, 600 (2016).
11. L. Cui, H. Hu, S. Yu, Q. Yan, Z. Ming, Z. Wen and N. Lu, *J. Netw. Comput. Appl.* **103**, 119 (2018).
12. J. Tang, R. Zhang, Y. Yao, F. Yang, Z. Zhao, R. Hu and Y. Yuan, *Physica A* **513**, 477 (2019).
13. M. Newman, *Phys. Rev. E* **74**, 036104 (2006).
14. A. Clauset, M. E. J. Newman and C. Moore, *Phys. Rev. E* **70**, 066111 (2004).
15. J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen and N. Glance, Cost-effective outbreak detection in networks, in *ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining* (San Jose, California, USA, 2007), pp. 420–429.
16. L. C. Freeman, *Soc. Netw.* **1**, 215 (1979).
17. M. E. J. Newman, *Soc. Netw.* **27**, 39 (2003).
18. Y. Du, C. Gao, Y. Hu, S. Mahadevan and Y. Deng, *Physica A* **399**, 57 (2014).
19. S. Brin and L. Page, *Comput. Netw. ISDN Syst.* **30**, 107 (1998).
20. L. L, Y. C. Zhang, C. H. Yeung and T. Zhou, *PLOS ONE* **6**, 1 (2011).
21. T. Cao, X. Wu, S. Wang and X. Hu, *Exp. Syst. Appl.* **38**, 13128 (2011).

22. T. Bian and Y. Deng, *Chaos Solitons Fractals* **103**, 101 (2017).
23. S. Pei, L. Muchnik, J. S. A. Jr, Z. Zheng and H. A. Makse, *Sci. Rep.* **4**, 5547 (2014).
24. Q. Jiang, G. Song, G. Cong, Y. Wang, W. Si and K. Xie, Simulated annealing based influence maximization in social networks, in *Proc. Twenty-Fifth AAAI Conf. Artificial Intelligence, AAAI'11* (AAAI Press, 2011), pp. 127–132.
25. J. Tang, R. Zhang, Y. Yao, Z. Zhao, P. Wang, H. Li and J. Yuan, *Knowl.-Based Syst.* **160**, 88 (2018).
26. C. Borgs, M. Brautbar, J. Chayes and B. Lucier, Maximizing social influence in nearly optimal time, in *Proc. Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'14* (Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2014), pp. 946–957.
27. H. T. Nguyen, M. T. Thai and T. N. Dinh, Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks, in *Proc. 2016 Int. Conf. Management of Data, SIGMOD'16* (ACM, New York, NY, USA, 2016), pp. 695–710.
28. Y. Tang, Y. Shi and X. Xiao, Influence maximization in near-linear time: A martingale approach, in *ACM SIGMOD Int. Conf. Management of Data*, 2015.
29. S. Lin, Q. Hu, G. Wang and P. S. Yu, Understanding community effects on information diffusion, in *Advances in Knowledge Discovery and Data Mining* (Springer International Publishing, Cham, 2015), pp. 82–95.
30. Y. Wang, G. Cong, G. Song and K. Xie, Community-based greedy algorithm for mining top-k influential nodes in mobile social networks, in *Proc. 16th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, KDD'10* (ACM, New York, NY, USA, 2010), pp. 1039–1048.
31. S. K. P. Suman Kundu, *Inf. Sci.* **316**, 107 (2015).
32. K. Rahimkhani, A. Aleahmad, M. Rahgozar and A. Moeini, *Exp. Syst. Appl.* **42**, 1353 (2015).
33. J. Shang, S. Zhou, X. Li, L. Liu and H. Wu, *Knowl.-Based Syst.* **117**, 88 (2017).
34. J. Shang, H. Wu, S. Zhou, J. Zhong, Y. Feng and B. Qiang, *Physica A* **512**, 1085 (2018).
35. U. N. Raghavan, R. Albert and S. Kumara, *Phys. Rev. E* **76**, 036106 (2007).
36. R. C. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, in *IEEE Int. Conf. Micro Machine and Human Science* (Nagoya, Japan, 1995), pp. 39–43.
37. J. Kennedy and R. C. Eberhart, A discrete binary version of the particle swarm algorithm, in *IEEE Int. Conf. Systems, Man, and Cybernetics. Computational Cybernetics and Simulation* (Orlando, FL, USA, 1997), pp. 4104–4108.
38. Y. Shi and R. Eberhart, A modified particle swarm optimizer, in *IEEE Int. Conf. Evolutionary Computation Proc. 1998. IEEE World Congress on Computational Intelligence* (Anchorage, AK, USA, 1998), pp. 69–73.
39. R. Liu, J. Li, J. Fan, C. Mu and L. Jiao, *Eur. J. Oper. Res.* **261**, 1028 (2017).
40. J. Meza, H. Espitia, C. Montenegro, E. Gimnez and R. Gonzlez-Crespo, *Appl. Soft Comput.* **52**, 1042 (2017).
41. R. Guimer, L. Danon, A. Dazguilera, F. Giralt and A. Arenas, *Phys. Rev. E* **68**, 065103 (2003).
42. S. Gregory, Finding overlapping communities using disjoint community detection algorithms, in *Complex Networks: Results of the 2009 Int. Workshop on Complex Networks (CompleNet 2009)*, eds. S. Fortunato, G. Mangioni, R. Menezes and V. Nicosia (Springer, Berlin, Heidelberg, 2009), pp. 47–61.
43. F. Lu, W. Zhang, L. Shao, X. Jiang, P. Xu and H. Jin, *J. Netw. Comput. Appl.* **86**, 15 (2017).
44. J. Leskovec, J. Kleinberg and C. Faloutsos, *ACM Trans. Knowl. Discov. Data* **1** (2007).