Knowledge-Based Systems 160 (2018) 88–103

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Maximizing the spread of influence via the collective intelligence of discrete bat algorithm

Jianxin Tang[a,b], Ruisheng Zhang[*,a], Yabing Yao[a], Zhili Zhao[a], Ping Wang[a], Huan Li[a], Jinliang Yuan[a]

[a] School of Information Science and Engineering, Lanzhou University,Lanzhou, Gansu 730000, China
[b] School of Computer and Communication, Lanzhou University of Technology, Lanzhou, Gansu 730050, China

ARTICLE INFO

Keywords:
Social network
Influence maximization
Metaheuristic
Discrete bat algorithm
Collective intelligence

ABSTRACT

Influence maximization aims to select a small set of *k* influential nodes to maximize the spread of influence. It is still an open research topic to develop effective and efficient algorithms for the optimization problem. Greedy-based algorithms utilize the property of "submodularity" to provide performance guarantee, but the computational cost is unbearable especially in large-scale networks. Meanwhile, conventional topology-based centrality methods always fail to provide satisfying identification of influential nodes. To identify the *k* influential nodes effectively, we propose a metaheuristic discrete bat algorithm (DBA) based on the collective intelligence of bat population in this paper. According to the evolutionary rules of the original bat algorithm (BA), a probabilistic greedy-based local search strategy based on network topology is presented and a *CandidatesPool* is generated according to the contribution of each node to the network topology to enhance the exploitation operation of DBA. The experimental results and statistic tests on five real-world social networks and a synthetic network under independent cascade model demonstrate that DBA outperforms other two metaheuristics and the Stop-and-Stair algorithm, and achieves competitive influence spread to CELF (Cost-Effective Lazy Forward) but has less time computation than CELF.

1. Introduction

Social networks have emerged as important platforms for information dissemination. Each individual in the network accumulates and exerts particular influence, which always results in the changes of an individual's thoughts, feelings, attitudes, or behaviors [1], to surrounding neighbors. Influence spread activities, such as product promotion, rumor mongering, opinion dissemination and even disease contagion, etc., can be solved by the "word-of-mouth" effect of viral marketing. Take the product promotion for example, a company plans to promote an innovative product to the market with a limited budget, a feasible way is to select a small set of influential individuals to trial the product freely. The expected result is the triers recommend the product to surrounding friends and family members iteratively through the power of "word-of-mouth" effect so that the marginal gain is maximal. Generally, this recommend procedure can be regarded as influence maximization, which was formalized into an algorithmic problem by Domingos and Richardson [2] firstly in terms of network perspective.

Influence maximization aims to select a small set of *k* influential nodes as a seed set so that these "superspreaders" can maximize the influence spread. Much attention has been attracted from researchers of different fields due to its potential commercial value and social value to network diffusion analysis as well as practical applications, such as viral marketing [3,4], critical infrastructure monitoring [5] as well as the prediction and control over opinion outbreaks [6,7], etc. Moreover, extended influence maximization problems, including topic-aware influence maximization (TIM), location-aware influence maximization (LIM) and competitive influence maximization (CIM), have been arousing the interest of researchers in recent years. According to TIM model, the influence spread between individuals is always computed by their topic preference learned from history activities on a social network. Considering the different influence strength of each topic, Chen et al. [8] developed a preprocessing-based MIS strategy for topic-aware influence maximization. As to LIM model, Li et al. [9] selected the top-*k* users having the highest influence upon a group of audience in a specified region by considering the spatial context. More recently, Li et al. [10] proposed an efficient community-based seeds selection (CSS) algorithm to approximate the influence spread on targeted users who have geographical preferences on queries. Driven by the precise location-aware advertising, Guo et al. [11] made the first attempt to

* Corresponding author.
E-mail address: zhangrs@lzu.edu.cn (R. Zhang).

https://doi.org/10.1016/j.knosys.2018.06.013
Received 8 January 2018; Received in revised form 28 June 2018; Accepted 30 June 2018
Available online 07 August 2018
0950-7051/ © 2018 Elsevier B.V. All rights reserved.

transplant the concept of influence maximization from social-aware advertising to location-aware advertising. They developed an expansion-based framework to enumerate the most influential trajectories to maximize the expected influenced audience based on the Singapore EZLink dataset. According to the CIM, multiple information factors diffuse in a social network at the same time. Li et al. [12] studied the dominated competitive mechanism and proposed a CIC-M model to enable multiple information to diffuse in the network simultaneously with the time-decay constraint. In addition, Marone and Makse [13] argued that the whole frame of interconnections in complex networks lies on a minimal set of structural influential nodes, and proposed the Collective Influence algorithm to identify the optimal minimum seed set for influence maximization by mapping the problem onto optimal percolation. Khomami et al. [14] proposed a learning automata approach to identify the minimum positive influence dominating seed set for influence maximization, but this algorithm suffers from the problem of improper parameter settings.

Additionally, the effectiveness and efficiency are the key factors of an algorithm for solving influence maximization problem. Kempe and Kleinberg [15] were the first to formulate influence maximization as a discrete optimization problem, and proved that the problem is NP-hard under independent cascade (IC) model and linear threshold (LT) model. Meanwhile, a hill-climbing greedy algorithm, which can achieve an approximation guarantee with errors bounded at $(1 - 1/e - \epsilon)$ to the optimal solution based on submodularity property, was proposed to solve the optimization problem. However, the simple greedy algorithm has to conduct tens of thousands Monte-Carlo simulations to estimate the influence spread of a given node set approximately, which is time consuming and not scalable to large-scale networks. To improve the efficiency of the simple greedy algorithm, successive improvements mainly on two topics were proposed in recent years. Some researchers tried to reduce the number of Monte-Carlo simulations, including the notable CELF [5] and CELF++ [16]. However, traversing tens of thousands subgraphs generated by each simulation takes up a huge amount of time, which seriously reduces the applicability for large-scale networks. Other researchers made efforts to construct more effective evaluation models [17,18] to estimate the expected influence spread of given node set and present more efficient heuristic algorithms [6,19,20] to optimize the evaluation functions. As emphasized in [21,22], there are two challenges in solving influence maximization problem. The first one is estimating influence spread of given node set approximately, which is proved to be #P-hard. The second one is providing effective and efficient algorithms to select a subset of k influential nodes to maximize the influence spread. Existing algorithms tend to suffer from the problems of large amount of computational time, low solution accuracy or huge memory cost especially with the increasing size of social networks. Therefore, it is still an open and thriving research topic to design effective and efficient algorithms for the influence maximization problem in large-scale networks.

In this paper, we focus on providing an effective influential nodes identifying algorithm by taking account of network topology comprehensively. A metaheuristic discrete bat algorithm (DBA) is proposed for the influence maximization problem based on the collective intelligence of bat population. The mainly contributions achieved in this paper include the following three aspects:

- Discrete evolutionary rules are designed for the metaheuristic bat algorithm according to network structural characteristics and a discrete bat algorithm is proposed for influence maximization problem.
- A probabilistic greedy-based local search strategy based on network topology is designed for the further exploitation of optimal solution confined to one-hop area of each bat in the population of each generation.
- A *CandidatesPool* maintaining potential influential nodes is generated for the random walk strategy according to the contribution of each node's degree centrality and its closeness centrality to the network topology.

To the best of our knowledge, this is the first time that the bat algorithm is adopted to tackle with influence maximization problem. The experimental results prove that the DBA is effective compared with state-of-the-art algorithms.

The rest of this paper is organized as follows. Section 2 introduces the related work on influence maximization. Related definitions of influence maximization and the independent cascade model are described in Section 3. Section 4 proposes the discrete bat algorithm for influence maximization. The experimental results and statistical hypothesis tests are provided in Section 5. Section 6 concludes this paper with future work.

2. Related work

To mine the commercial value of customers, Domingos and Richardson [2] studied the expected profit from the customers of viral marketing firstly in a network perspective and modeled the influence maximization problem as a Markov random field. Subsequently, Kempe and Kleinberg [15] formulated influence maximization as a discrete combinatorial optimization problem, and proved it to be NP-hard under IC and LT model. Meanwhile, a hill-climbing greedy algorithm, which can guarantee $(1 - 1/e - \epsilon)$ approximation of the optimal solution, was proposed. However, to estimate the influence spread of a given node set approximately, tens of thousands Monte-Carlo simulations have to be conducted, which is high computational cost and not scalable to large-scale networks. To improve the efficiency of the simple greedy algorithm, new enhancements were proposed. Leskovec et al. [5] demonstrated that the objective function of influence spread exhibits the property of "submodularity", and they proposed the CELF algorithm by maintaining a priority queue of the influence spread of each node to reduce the number of Monte-Carlo simulations. To further reduce the marginal gain computation cost of the Monte-Carlo simulations in CELF, Goyal et al. [16] proposed the CELF++ algorithm, which can achieve 30–35% performance improvement in terms of execution time, but yields to the effectiveness of CELF. Both algorithms perform well on identifying the seed nodes at small propagation probability, such as the frequently-used $p = 0.01$ under IC model, and are scalable to large-scale networks within a relaxed time budget. However, the two algorithms tend to need huge computation once the propagation probability is large, such as $p = 0.05$ under IC model, or the node degree distribution of the network is normally distributed. By removing all edges not for influence propagation from the original network, Chen et al. [19] proposed the MixedGreegy algorithm which outperforms traditional centrality-based methods. Meanwhile, the authors pointed out that it may be a good choice to develop heuristic algorithms based on network topology. Then they proposed the efficient SingleDiscount (SD) and DegreeDiscount (DD) algorithms. To enable greedy algorithms to perform well in large networks, community-based greedy algorithms were proposed to reduce the computational time of the simple greedy algorithm. Wang et al. [23] took into account the information diffusion degree of edge weight to detect the communities of large-scale networks firstly, then the dynamic programming method was adopted to select influential nodes among the communities, but the heuristic method cannot provide efficient identification of influential nodes. Considering the networks with clear community structure, Zhang et al. [24] detected the communities firstly using the information transfer probability between any pair of nodes, then the k-medoid clustering algorithm was adopted to identify the k influential nodes. Shang et al. [25] proposed a community-based framework for influence maximization (CoFIM). According to the framework, seed nodes were expanded firstly to get second-order seeds among the communities, then the evaluation of the influence spread within each community was conducted independently to select the most influential nodes iteratively. The experiments showed that CoFIM can identify the k most influential nodes in an efficient way, yet the parameter setting in CoFIM is sensitive to different networks.

Novel estimation mechanisms for node influence evaluation are also of great importance to solve the influence maximization in large-scale networks. Lu et al. [26] suggested a scheme that recursively estimates the influence spread using reachable probabilities from node to node, and adopted the greedy framework to select the nodes with the most recursive influence estimation value. Han et al. [27] argued that the timeliness decay of influence, acceptance ratio and propagation breadth are three important factors of influence maximization. So the authors utilized the ideology of structural hole spanners to measure the contribution of each influential node in maximizing the influence breadth, and proposed the ICOT model to evaluate node influence propagation and select the potential influential nodes those covering as more communities as possible.

Different with greedy algorithms that select the k most influential nodes sequentially, centrality heuristics based on network topology, including degree centrality [15] and *K-shell* decomposition [28], etc., identify influential nodes simultaneously once the network topology is given. However, these methods always suffer from low and unguaranteed accuracy especially in scale-free networks since the nodes with the highest degree or *K-shell* value tend to be highly clustered, which will result in overlap phenomenon of influence spread. Novel heuristics based on network topology were proposed to enhance the identification of influential nodes. To improve the performance of *K-shell* decomposition on identifying multiple influential nodes, Yeruva et al. [29] utilized the Pareto front function to estimate the spread of node influence, and took into account of the metrics including node's out-degree, in-degree and high degree to obtain the Pareto front optimal solution set for influence maximization. Kim et al. [30] proposed an effective pruning heuristic method RWP based on *Random Walk* and *Rank Merge* schemes to prune out uninfluential nodes in the network. However, the influence spread achieved by RWP is always suboptimal compared to state of the art algorithms though the method can reduce an amount of computation for evaluating the influence spread of node set.

The vitality of random sampling theory in solving influence maximization has been proved by recent studies. To estimate the expected influence spread of a set of seed nodes more effectively and efficiently in large-scale networks, Borgs et al. [31] suggested a novel reverse influence sampling (RIS) mechanism based on uniform sampling theory for influence maximization problem, and the algorithm can obtain a near-optimal approximation factor of $(1 - 1/e - \epsilon)$ within nearly optimal time. To optimize the number of samples needed to be generated and the sampling scheme, some modified methods were proposed to overcome the deficiencies of the basic RIS algorithm. TIM+ [32] and SKIM [33] were proposed to select seed nodes incrementally using a concept of combined reachability sketch, but the two algorithms need huge amounts of memory cost for the generated subgraphs. Two novel sampling frameworks named SSA and D-SSA, which are up to 1200 times faster than the IMM [18], were proposed by Nguyen [34] recently. However, those sample-based algorithms suffer from the problem of having to generate tens of thousands subgraphs, which consume huge memory cost, to evaluate the expected influence spread of nodes approximately. Thus, there are still challenges and bottlenecks such as reasonable computational time, guaranteed accuracy, and low memory consumption to design effective and efficient algorithms for influence maximization in large-scale networks.

Metaheuristic algorithms, mimicking the behaviors of biotic population or characteristic transformation of physical phenomena, have been employed as powerful methods to solve the influence maximization in the last few years. Simulated annealing (SA) algorithm was applied to influence maximization by Jiang et al. [35] for the first time. The experimental results showed that SA runs faster than the simple greedy algorithm by 2–3 orders of magnitude. Sankar et al. [36] proposed a bee algorithm based on the *waggle dance* strategy of bee colony and verified the performance on influence propagation of the proposed algorithm on a Twitter dataset. Gong et al. proposed a discrete particle swarm optimization (DPSO) [20] to identify top-k influential nodes, but the local search strategy adopted in DPSO tends to lead the algorithm to suboptimal solution. Meanwhile, Gong et al. [37] also proposed a memetic algorithm for community-based influence maximization, which outperforms traditional heuristics in multiple influential nodes identification. Cui et al. [38] proposed a degree descending search evolutionary (DDSE) algorithm for influence maximization. According to the framework of DDSE, the discrete operations including mutation, crossover and selection involved in the differential evolutionary (DE) can identify influential nodes efficiently. A common superiority of metaheuristic algorithms is that these methods avoid generating tens-of-thousands subgraphs or traversing Monte-Carlo simulations to compute the influence by optimizing the evaluation function to estimate the expected influence spread of given node set. Moreover, extensive experimental results demonstrated that it is a promising way to solve influence maximization problem by taking the advantages of metaheuristic optimization algorithms.

As a novel member of the metaheuristic algorithms, the bio-inspired bat algorithm (BA) that mimics the echolocation behavior of microbats while preying in the darkness was proposed by Yang [39] in 2010. The validation based on benchmark functions proved that BA is superior to genetic algorithm (GA) and particle swarm optimization (PSO), etc., in tackling with complex optimization problems [40–42]. Great attention and interest has been attracted from researchers and engineering experts since its first implementation. Saji and Riffi [43] extended the application of BA and proposed a discrete bat algorithm to solve the well-known traveling salesman problem (TSP). The experimental comparisons on 41 symmetric instances of TSP showed that the discrete bat algorithm outperforms the particle swarm optimization and an improved cuckoo search algorithm. More recently, Osaba et al. [44] omitted the parameter "frequency" and proposed an improved discrete bat algorithm (IBA), which can be seen as a special case of classical PSO, for symmetric and asymmetric TSP problems. The IBA outperformed other metaheuristics including the firefly algorithm according to the experimental results. To evaluate the optimal thresholds using Otsu's thresholding tool for grayscale image segmentation applications, Satapathy et al. [45] proposed a novel chaotic bat algorithm (CBA) to maximize the between-class variance function in Otsu technique. To solve the combined economic/emission dispatch problem with power flow constraints, Liang et al. [46] proposed a multiobjective hybrid bat algorithm, in which learning strategy and sorting methods were adopted to improve the population diversity and the global optimal Pareto Optimal Front. In this paper, we employ the metaheuristic bat algorithm and present a discrete bat algorithm (DBA) by modifying the evolutionary rules of the original bat algorithm based on network topology to provide effective identification of influential nodes for the influence maximization problem in large-scale networks.

3. Preliminaries

3.1. Influence maximization

According to the mathematical model formatted in [15], the influence maximization problem can be generally defined as follows.

Definition 1. Let $G = (V, E, W)$ be a graph, V represents the node set with $|V| = n$ nodes and E represents the edge set with $|E| = m$ edges, where $E \subseteq \{(u, v)|u, v \in V\}$. 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 or learned with respective spreading rules problem-specially.

Definition 2. Given a graph $G = (V, E, W)$ and a positive integer k $(1 \leq k \ll n)$, the target of influence maximization is to select k most influential nodes as seed set S so as to the expected number of triggered nodes, denoted as influence spread $\sigma(S)$, is maximal under a given cascading propagation model.

Require: Objective function $f(x)$, $x = (x_1, x_2, \ldots, x_d)^T$
Initialize the bat population x_i $(i = 1, 2, \ldots, N)$, and v_i
Define pulse frequency f_i at x_i
Initialize pulse rate r_i and the loudness A_i
while $t <$ Max number of iterations **do**
    Generate new solutions by adjusting frequency, and updating
    velocities and locations/solutions according to Eqs. (**??**)–(**??**)
  if $rand > r_i$ **then**
    Select a solution among the best solutions
    Generate a local solution around the selected best solution
  end if
  Generate a new solution by flying randomly
  if $rand < A_i$ && $f(x_i) < f(x_*)$ **then**
    Accept the new solutions
    Increase r_i and reduce A_i
  end if
  Rank the bats and find the current best x_*
end while
Postprocess results and visualization

Algorithm 1. Bat algorithm.

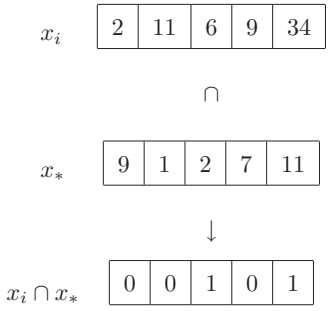

Fig. 1. An illustration of operator `` ∩ '' on two location vectors.

$$S^* = \arg\max_{S \subseteq V, |S|=k} \sigma(S) \tag{1}$$

where, $\sigma(S)$ is the expected number of nodes that are influenced by S in the given graph G, and S^* is the best seed set returned by the critical function.

3.2. Propagation model

Propagation probability p in stochastic cascade models describes the tendency of an inactive individual to be affected by its direct active neighbors. The IC model [15], which has been widely adopted in previous works, is a classic probability model mimicking information diffusion in social networks.

According to IC model, each node has only two states, either active or inactive, and inactive nodes can be influenced and changed into the active nodes, but not vice versa. For an active node u at step t, it has only one chance to activate each of its direct inactive neighbors v and successes with a probability p_{uv} that is associated with edge $(u, v) \in E$. Whether the activation is succeed or not, u will no more attempt to activate v in the following steps. If node v is activated by u successfully, then v will be active and has one chance to activate its direct inactive neighbors in step $t + 1$. The diffusion process stops if no node is activated at step T and returns the $\sigma(S)$ comprising all of the active nodes.

In this paper, we adopt the IC model as the influence cascade model at propagation probability $p = 0.01$ intensively to validate the performance of the proposed discrete bat algorithm for influence maximization.

4. Proposed method

4.1. Bio-inspired bat algorithm

Bats are fascinating animals for they are the only mammals with wings in the nature. Studies led by biologists found that most microbats use a type of sonar, called echolocation, to detect prey, avoid obstacles and locate their roosting crevices in the dark. According to the statements [39], the ideology of BA can be described as follows: a population of N bats associated with velocity $v_i^t (1 \le i \le N)$ and location x_i^t respectively search in a d-dimensional solution space, at time stamp t, they 'fly' through collective intelligence to the prey x_*, which locates at a fixed place, until they find the prey. By idealizing the behavior rules of microbats, the mathematical evolutionary rules for location x_i^t and velocity v_i^t are formulated as follows:

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{2}$$

$$v_i^{t+1} = v_i^t + (x_i^t - x_*)f_i \tag{3}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{4}$$

where, each bat in the population associates with an initial frequency which is drawn uniformly from $[f_{min}, f_{max}]$. $\beta \in [0, 1]$ is drawn randomly from a uniform distribution and x_* is the global best location of current generation, which can be located after comparing all the solutions provided by the N bats.

To provide an effective mechanism to coordinate the global exploration and local exploitation of BA, parameters including loudness A_i and pulse emission rate r_i associated with each bat are pruned dynamically according to Eqs. (5) and (6) during the evolutionary generations.

$$A_i^{t+1} = \alpha A_i^t \tag{5}$$

$$r_i^{t+1} = r_i^0[1 - exp(-\gamma t)] \tag{6}$$

where, α and γ are two constants. Generally, the loudness A_i usually decreases once a bat has found its prey, meanwhile, the rate of pulse emission r_i increases with bat i approaching to the prey, so we have $0 < \alpha < 1$, $\gamma > 0$, and

$$A_i^t \to 0, \quad r_i^t \to r_i^0, \quad as\ t \to \infty \tag{7}$$

Extensive simulations show that metaheuristic algorithms tend to

Require: Network $G = (V, E)$, maximum of generations g_{max}, bat population size N, frequency bounds f_{min} and f_{max}, influence evaluation model $LIE(\cdot)$.

Initialize iterator $g = 0$
Initialize pulse rate vector r with number drawn randomly from (0, 1)
Initialize location vector x based on high degree centrality heuristic
Initialize velocity vector v with 0 or 1 drawn randomly from {0, 1}
Select out initial best location vector x_* according to the LIE of each x_i
while $g < g_{max}$ **do**
    for each bat $x_i \in x$ **do**
        Update the velocity vector v_i according to Eq. (**??**)
        Update the position vector x_i according to Eq. (**??**)
        if $rand > r_i$ **then**
            $x_i \leftarrow LocalSearch(x_i, G)$
        end if
        $x_i^{new} \leftarrow RandomWalk(x_i, CandidatesPool, k)$
        if $LIE(x_i^{new}) > LIE(x_i)$ && $rand < A_i$ **then**
            Accept x_i^{new}
        end if
        Update $A_i \leftarrow \alpha A_i$
        Update $r_i \leftarrow r_i^0[1 - exp(-\gamma g)]$
    end for
    Select out the best global location bat x_* of the current generation
    $g \leftarrow g + 1$
end while
Ensure: x_* as the best seed set S

Algorithm 2. Framework of DBA for influence maximization.

Require: bat location x_i.
$d_{x_i} \leftarrow Degree(x_i)$
$x_i^{'} \leftarrow Order(x_i, d_{x_i})$
for each element $x_{ij} \in x_i^{'}$ **do**
  $Nei\_set \leftarrow N_{x_{ij}}^{(1)}$
  $index \leftarrow 1$
  $len \leftarrow Length(Nei\_set)$
  while $index \leq len$ **do**
    if $rand() > 0.5$ **then**
      $x_i^{'} \leftarrow Replace(x_{ij}, Nei\_set)$
      if $LIE(x_i^{'}) > LIE(x_i)$ **then**
        $x_i \leftarrow x_i^{'}$
      end if
    end if
    $index \leftarrow index + 1$
  end while
end for
Ensure: the newly local best seed set x_i.

Algorithm 3. Local search strategy *LocalSearch*(x_i, G).

suffer from some drawbacks in dealing with multimodal optimization problems [39]. One major enhancement is providing local search strategy to execute exploitation operation to avoid the population being trapped into local optima prematurely. For the original local search part of BA, once a solution, i.e., a bat in the population, is selected among the current best solutions, a new solution for the bat is generated locally using random walk and accepted with probabilities:

$$x_{new} = x_{old} + \epsilon A^t \tag{8}$$

where ϵ is a random number drawn uniformly from $[-1, 1]$, and $A^t = <A_i^t>$ is the average loudness of all the bats at time stamp t. The basic steps of the bat algorithm can be summarized as the pseudo code shown in Algorithm 1.

4.2. Discrete bat algorithm

4.2.1. Encoding the bat individual

The original BA was developed for continuous combinational optimization problems. To tackle with the influence maximization problem in a discrete network structure, we redesign an encoding mechanism for bat individual to propose the discrete bat algorithm (DBA) according to the characteristics of network topology. As discussed in Section 1, the influence maximization problem aims to find a seed set of k most influential nodes to maximize the spread of influence. To solve the influence maximization problem by using the bat algorithm, each bat individual in the bat population can be represented by k potential candidate nodes, i.e., the location vector of each bat individual is composed of k node units. Each unit in the location vector will be updated according to the collective intelligent rules of the discrete bat algorithm until the termination condition is satisfied, then the best bat individual can be treated as the targeted seed set. Therefore, the influence maximization problem can be solved by discrete bat algorithm.

Location encoding: Given the nodes in a network are numbered uniquely by positive numbers, so each bat in the population can be encoded by k potential influential nodes from the network, i.e., the location of bat i can be defined as a k-dimensional vector $x_i = (x_{i1}, x_{i2}, ..., x_{ik})$ (where $i = 1, 2, ...,N$) , which is similar to that of DPSO. For example, bat individual $x_i = (2, 11, 6, 9, 34)$ represents a candidate seed set consisting of $k = 5$ nodes for influence maximization, each element x_{ij} $(j = 1, 2, ... k)$ in x_i is drawn from the node set V of a network and acts as a potential influential seed node.

Velocity encoding: Similar to the location encoding mechanism, the velocity of each bat is encoded by k decisioning constants consisted of 0 and 1, i.e., the velocity vector of bat i is defined as $v_i = (v_{i1}, v_{i2}, ..., v_{ik})$, where v_{ij} $(j = 1, 2, ... k)$ is a decisioning factor for the corresponding seed node x_{ij} in x_i. To be more specific, if $v_{ij} = 1$, the corresponding node x_{ij} needs to be replaced with another candidate node drawn from the network, otherwise, $v_{ij} = 0$ implies that the corresponding node can be reserved in the current seed set.

Table 1
Statistic characters of the six networks. $|N|$ 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, C represents the average clustering coefficient, and AC represents the assortativity coefficient.

Networks	$\|N\|$	$\|E\|$	$\langle k \rangle$	$\bar{d}$	C	AC
HepTh	9877	25,998	5.264	5.945	0.600	0.268
PGP	10,680	24,316	4.554	7.486	0.440	0.238
SynRand	15,000	56,152	7.491	5.006	0.0005	−0.0025
CondMat	23,133	186,936	16.162	5.352	0.055	0.135
Deezer	54,573	846,915	31.038	3.609	0.071	0.172
Slashdot	77,360	905,468	23.409	4.024	0.087	−0.046

4.2.2. Discrete evolutionary rules

According to the encoding mechanisms for bat individual, newly evolutionary rules for bat location x_i^t and velocity v_i^t are redefined as the following Eqs. (9) and (10).

$$v_i^{t+1} \leftarrow H(v_i^t + (x_i^t \cap x_*) f_i) \tag{9}$$

$$x_i^{t+1} \leftarrow x_i^t \oplus v_i^{t+1} \tag{10}$$

where "$\cap$" is a logic operator similar to intersection operation to check whether there are common elements x_{ij} between x_i and x_*, that is, if there is a same node in x_* with x_{ij} in x_i, then the corresponding element in the result vector is set to 0, and 1 otherwise. As the illustration shown in Fig. 1, $x_i = (2, 11, 6, 9, 34)$ and $x_* = (9, 1, 2, 7, 11)$ are encoded by $k = 5$ influential nodes, respectively, so the corresponding "$\cap$"

Require: *CandidatesPool*, seed set size k and the coefficient β.
$index \leftarrow 1$
$x_i^{new} \leftarrow \Phi$
while $index \leq k$ **do**
  $temp \leftarrow Random(CandidatesPool, index)$
  if $temp \notin x_i^{new}$ **then**
    $x_i^{new} \leftarrow temp$
  end if
  $index \leftarrow index + 1$
end while
Ensure: the new bat location x_i^{new}.

Algorithm 4. Random walk strategy *RandomWalk*(x_i, *CandidatesPool*, k).

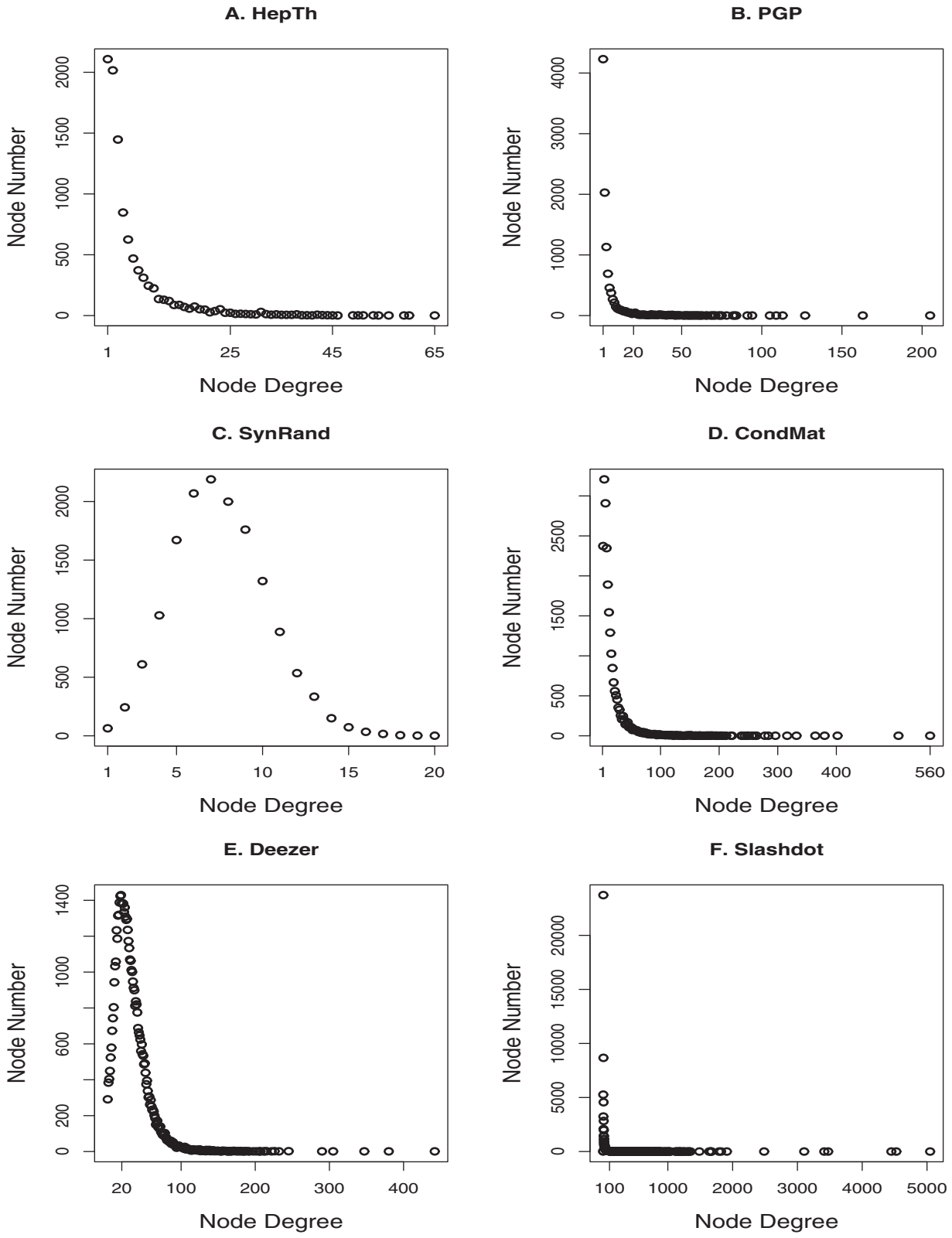

Fig. 2. Node degree distribution of the six networks.

operation result can be denoted as $(0, 0, 1, 0, 1)$ on the two location vectors.

In the right of Eq. (9), v_i^t and $x_i^t \cap x_*$ can be regarded as 'self-learning' part and 'collective-learning' part of each bat respectively, and f_i is the inertia weight of the latter part, which is adjusted dynamically with the bat approaching to the optima. Each bat in the population updates its velocity according to this form of collective intelligence, through which the population converges to global optimal solution.

$H(\cdot)$ is a decision function utilized to formalize bat's velocity vector once the addition operation is finished, and it is formulated by considering the two inertia coefficients of parts v_i^t and $x_i^t \cap x_*$. Given that v_i is the parameter, then $H(v_i)$ can be represented as

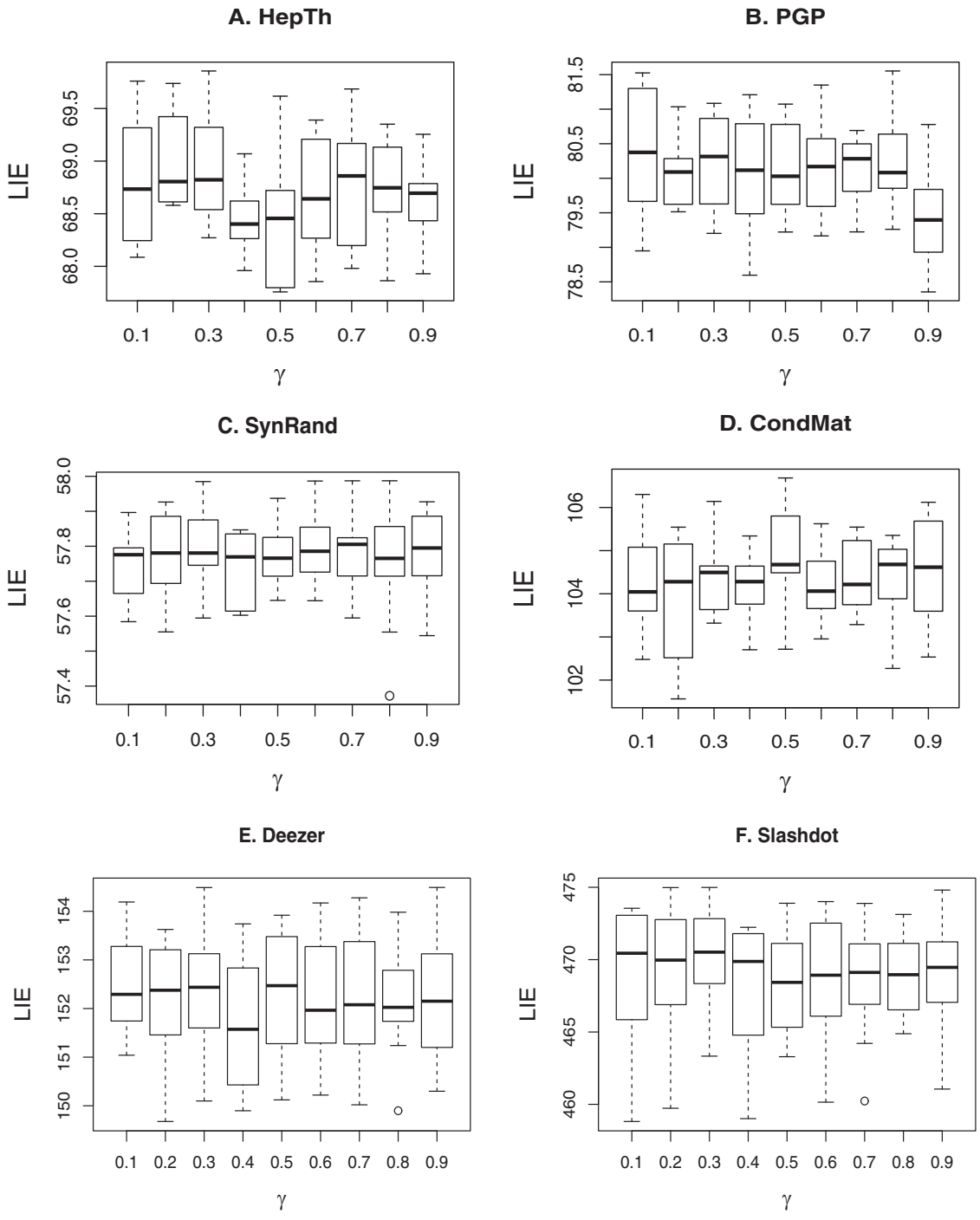

Fig. 3. Box-plots of the expected local influence spread estimation (*LIE*) of targeted 50 seed nodes with γ varying from 0.1 to 0.9 at propagation probability $p = 0.01$ in the six networks, respectively.

$H(v_i) = (h_1(v_{i1}),\ h_2(v_{i2}), ..., h_k(v_{ik}))$, where $h_j(v_{ij})$ $(1 \le j \le k)$ is defined as a threshold factor, as formulated in Eq. (11), according to the inertia coefficients of self-learning part and collective learning part in the right of Eq. (9).

$$h_j(v_{ij}) = \begin{cases} 0, & if \quad v_{ij} < \frac{1+(f_{max}-f_{min})}{2} \\ 1, & if \quad v_{ij} \ge \frac{1+(f_{max}-f_{min})}{2} \end{cases} \tag{11}$$

For example, if we fix the frequencies $f_{min} = 0$, $f_{max} = 2$ and $\beta = 0.65$, then the velocity vector v_i can be calculated as the following format $v_i = H((1, 1, 0, 0, 1) + (1, 1, 1, 0, 1)\beta) = H((1.65, 1.65, 0.65, 0, 1.65))$.

$= (1, 1, 0, 0, 1)$

Once the decision function $H(\cdot)$ returns the velocity vector v_i^{t+1} of bat i in the current generation, logic operator "⊕" is tasked with determining whether the current seed node x_{ij}^t in x_i^t should be kept or updated. Considering the updating rule of location, each element x_{ij}^{t+1} in x_i^{t+1} can be updated according to Eq. (12).

$$x_{ij}^{t+1} = \begin{cases} x_{ij}^t, & if \quad v_{ij}^{t+1} = 0 \\ Replace(x_{ij}^t,\ CandidatesPool), & if \quad v_{ij}^{t+1} = 1 \end{cases} \tag{12}$$

where, $Replace(\cdot)$ is a function that replaces element x_{ij}^t with a random node drawn from the *CandidatesPool*, which is generated for the random walk strategy of the discrete bat algorithm, and guarantees there is no

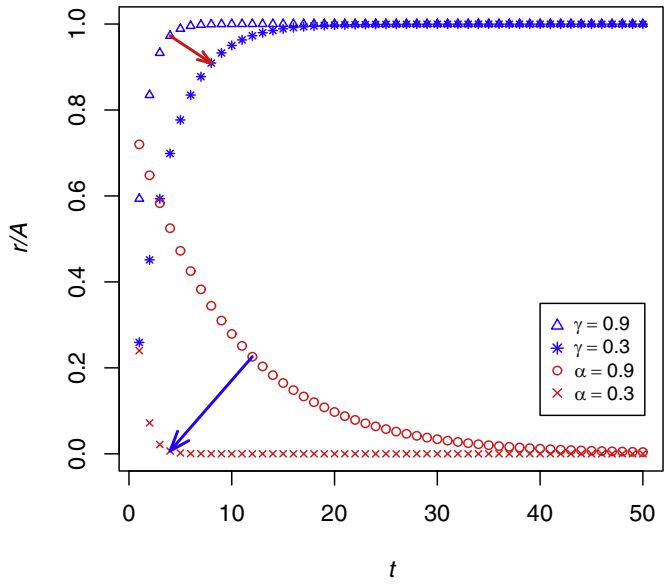

Fig. 4. Convergence trend of the pulse emission rate r and loudness A at 0.9 and 0.3 separately over the time stamp t.

repeated node in x_i after the replacement is finished. In other words, the k candidate influential nodes in the global best location x_* are updated iteratively until the terminal condition is satisfied.

4.3. DBA for influence maximization

4.3.1. Influence evaluation model

The function shown in Eq. (1) is a typical optimization problem, and it has been proved to be NP-hard by Kempe and Kleinberg [15]. Therefore, effective and efficient evaluation models to estimate the influence spread of given node set approximately are essential for the identification of influential nodes in large-scale networks. As addressed in [47], the influence of an individual from social networks decays with one's friendship delimitation, and the influence spread tends to be localized in a numbered friends of local area. Furthermore, Pei et al. [48] indicated that the sum of the nearest neighbors degree is a reliable local proxy for node influence, and the expected local influence spread of a node can be evaluated within its two-hop area especially when the complete global network structure is unavailable. According to the suggestion, a local influence estimator (*LIE*) function shown in Eq. (13) was modeled to approximate the influence spread within the two-hop area of an influential node. In addition, the advantage that *LIE* can provide reliable estimation of influence spread at small propagation probability was verified in [20].

$$LIE = \sigma_0(S) + \sigma_1^*(S) + \tilde{\sigma}_2(S) \tag{13}$$

In Eq. (13), $\sigma_0(S)$ is the number of initial influential nodes in seed set S, $\sigma_1^*(S)$ and $\tilde{\sigma}_2(S)$ are the expected influence spread of one-hop and two-hop area of the seed set S, respectively. For the *LIE* of one-hop area and two-hop area can be expressed according to the adjacency matrix of the k nodes abstracted from the network, then the *LIE* can be calculated by the following Eq. (14)

$$\begin{aligned} LIE &= k + \sigma_1^*(S) + \frac{\sigma_1^*(S)}{|N_S^{(1)} \setminus S|} \sum_{u \in N_S^{(2)} \setminus S} p_u^* d_u^* \\ &= k + \left(1 + \frac{1}{|N_S^{(1)} \setminus S|} \sum_{u \in N_S^{(2)} \setminus S} p_u^* d_u^*\right) \sum_{i \in N_S^{(1)} \setminus S} \left(1 \right. \\ &\quad \left. - \prod_{(i,j) \in E, j \in S} (1 - p_{i,j})\right) \end{aligned} \tag{14}$$

where $N_S^{(1)}$ and $N_S^{(2)}$ represent the one-hop and two-hop area of seed set S, respectively. p_u^* is a small constant propagation probability for the given diffusion model, and d_u^* is the number of out-edges of node u within $N_S^{(1)}$ and $N_S^{(2)}$.

According to Eq. (13), the identification of influential nodes is transformed into an optimization problem with the aim of selecting a seed set to maximize the *LIE* function fitness value. In this paper, we focus on providing an effective seed set identifying algorithm based on the *LIE* influence evaluation model. Thus, seeking for a set of influential nodes to maximize the fitness of *LIE* is the goal of our algorithm.

4.3.2. Framework of DBA for influence maximization

According to the original evolutionary rules of BA, the location and velocity updating rules are refereed as exploration factors to search global optimal solution. To avoid bats in the population searching blindly, a probabilistic greedy-based local search strategy and a random walk strategy based on the discrete network topology are presented for DBA in line with the original ideology of BA. Both the local search and random walk act as exploitation factors to improve the solution quality locally. The framework of DBA for influence maximization is given in Algorithm 2.

In the framework, we initialize the location vector x based on high degree centrality, i.e., each bat in the population is initialized by k nodes with the highest degree metric. Meanwhile, a random disturbance mechanism is adopted to diversify the initial population. Function $LIE(\cdot)$ calculates the expected local influence spread of seed set S according to Eq. (14). In addition, functions $LocalSearch(x_i, G)$ and $RandomWalk(x_i, CandidatesPool, k)$ are detailed in Sections 4.3.3 and 4.3.4, respectively.

4.3.3. Local search strategy

Local search procedure is an individual reinforcement operation which aims to find a better solution around the best one found so far. As stated in Section 4.1, local search strategy based on network topology plays a significant role in exploiting more influential nodes. Based on the discrete network topology, a probabilistic greedy-based local search is presented for the further exploitation of DBA. Firstly, the k nodes in the location vector of the selected bat are ranked in an ascending order by high degree metric to let the nodes with less degree value exploit more influential neighbors preferentially. Secondly, a probabilistic greedy-based local strategy given in Algorithm 3 is applied on each candidate node from the temporary bat location to replace the current node with the most influential node from its one-hop neighbor set while keeping the diversity of the bat population. That is, once a temporary bat location of current iterative generation is selected, we apply the local search strategy on the bat location subsequently to get a new locally optimal bat location for the evolution of next generation until the termination condition is satisfied.

In Algorithm 3, G is a network $G = (V, E)$, function $Degree(x_i)$ returns the degree value of each node in bat location x_i. Function $Order(x_i, d_{x_i})$ returns an ordered node set on the basis of high degree metric in an ascending order. Function $Length(Nei\_set)$ returns the size of the neighbor set Nei_set. $Replace(x_{ij}, Nei\_set)$ is adopted to replace x_{ij} with a neighbor from the one-hop area of x_{ij} and guarantees that there is no repeated nodes after the replacement is finished.

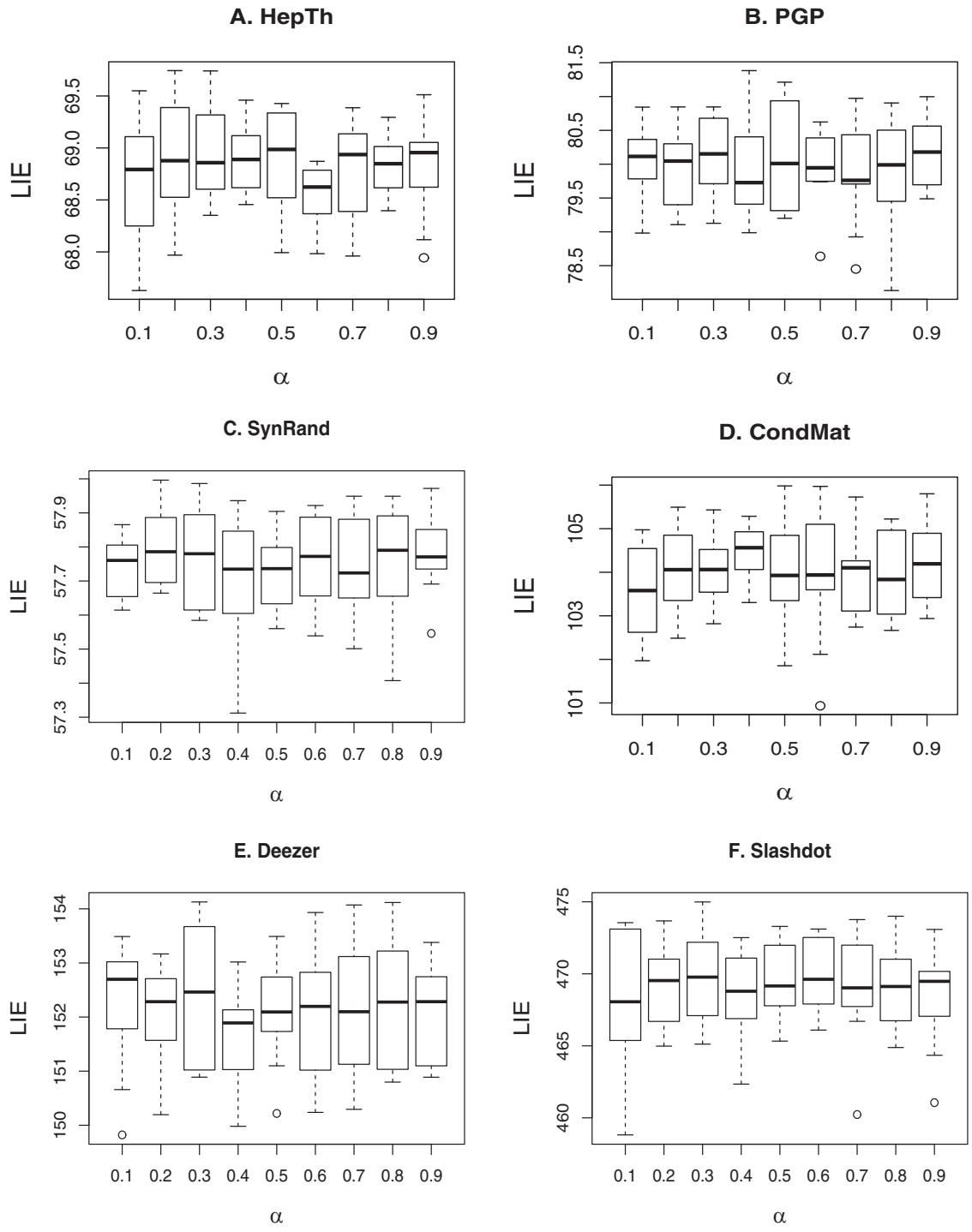

Fig. 5. Box-plots of expected local influence spread estimation (*LIE*) of target 50 seed nodes with α varying from 0.1 to 0.9 at propagation probability $p = 0.01$ in the six networks, respectively.

4.3.4. Random walk strategy

According to the evolutionary rules of the original BA, once a solution is selected, a new location for the bat is suggested to be generated locally using a random walk strategy. Different to the continuous optimization problems, the influence maximization problem is a discrete one, in other words, we cannot generate a new bat location according to Eq. (8) directly.

To generate a new location for the selected bat, we introduce a network topology-specially random walk strategy to avoid the bat exploiting blindly. Moreover, a *CandidatesPool* is generated according to the contribution of each node to the network topology. The mathematical formalization is given in Eq. (15).

$$C_i = \mu \frac{d_i}{\sum_{j=1}^{N} d_j} + \nu \frac{CC_i}{\sum_{j=1}^{N} CC_j} \tag{15}$$

where C_i is the contribution value of node i, which is calculated on two metrics including node degree centrality and its respective closeness centrality. The two parts on the right of Eq. (15) are the contribution rates of each node's degree centrality (DC) and closeness centrality (CC), respectively, μ and ν are two constant coefficients. We initialize the size of *CandidatesPool* with $\beta \cdot k$, i.e., β times of the targeted seed set

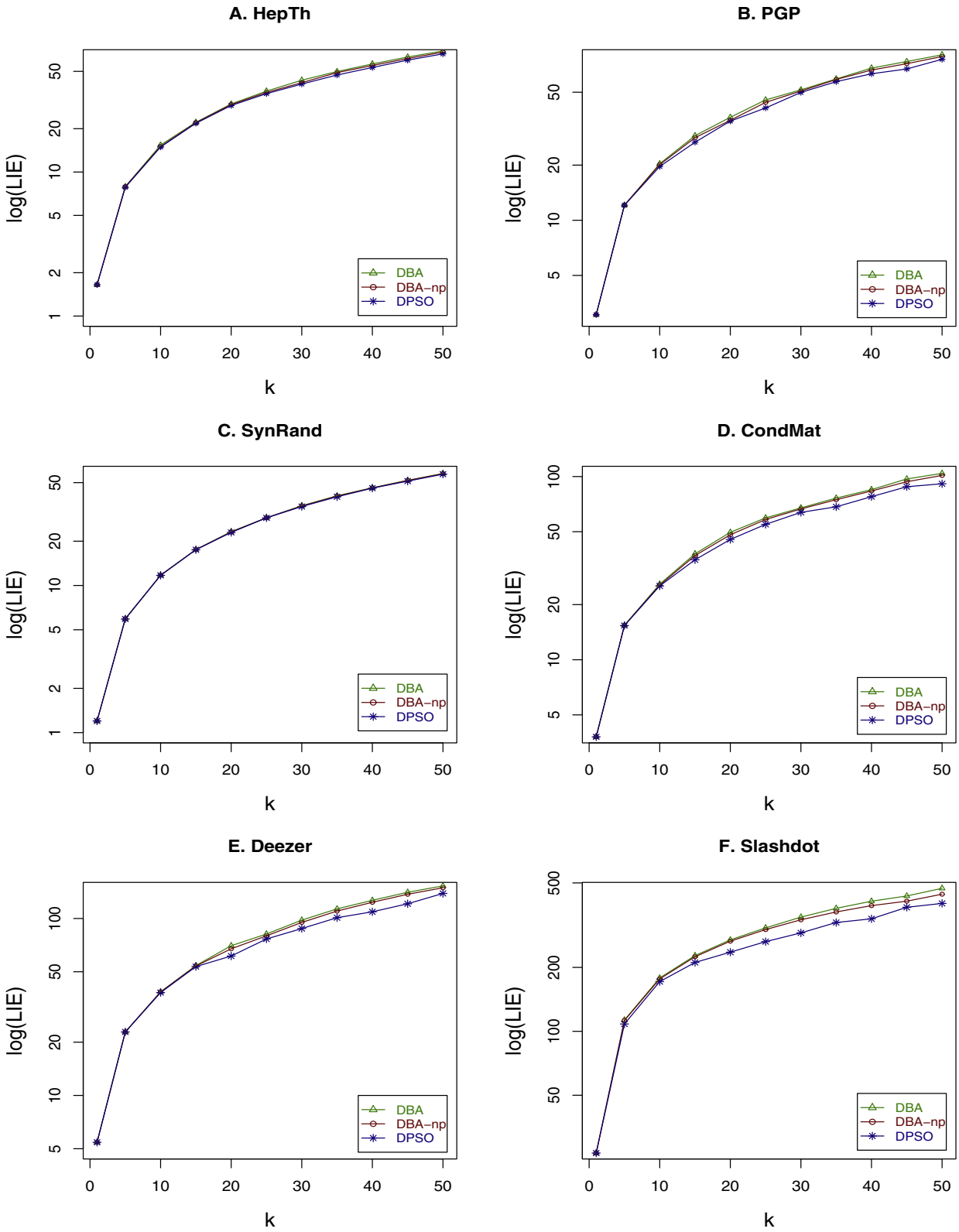

Fig. 6. Comparisons on optimizing the *LIE* function of the three metaheuristics DBA, DBA-np and DPSO at propagation probability $p = 0.01$ in the six networks.

size k. So $\beta \cdot k$ candidate nodes with the highest contribution value will be selected to fill up the *CandidatesPool*. The pseudocode of random walk is given in Algorithm 4, in which function *Random*(*CandidatesPool*, *index*) returns a node ID drawn randomly from the *CandidatesPool* as an element of the new bat x_i^{new} indexed by parameter *index*.

4.3.5. Computational complexity of DBA

According to the framework of DBA for influence maximization, the computational complexity of local search strategy is the major factor affecting the efficiency of DBA, of which the running time lies in the ordering operation and probabilistic greedy-based replacement. The ordering operation requires $O(k \cdot \log k)$, and the replacement requires

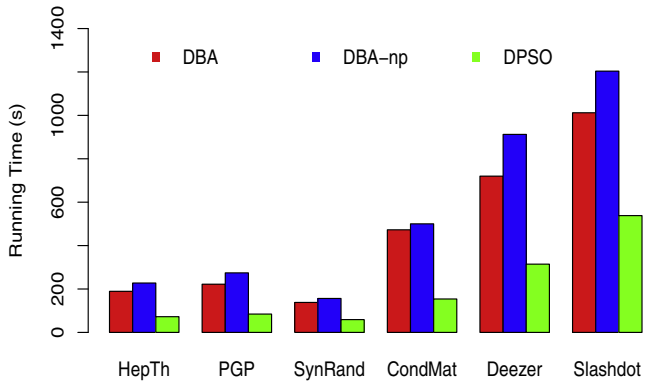

Fig. 7. Running time of three algorithms on optimizing the *LIE* function fitness with targeted seed set size $k = 50$ in the six networks.

$O(k{\cdot}\overline{D})$, where $\overline{D}$ is the average node degree of a network, thus, the whole time complexity of local search strategy is $O(k(\log k + \overline{D}))$. The degree-based initialization needs $O(N{\cdot}k)$. Updating velocity vector needs $O(k{\cdot}\log k{\cdot}N)$, and updating location vector needs $O(k{\cdot}N)$. In addition, random walk operation needs $O(k)$, and evaluating *LIE* needs $O(k{\cdot}\overline{D})$ operations. The others need one unit operation separately. Thus, the upper bound computational complexity of DBA is $O(N{\cdot}k{\cdot}(\log k + \overline{D}){\cdot}g_{max})$.

5. Experimental results and statistical tests

5.1. Preparation

To validate the performance of DBA for influence maximization problem, we conduct extensive experiments on six undirected networks including a synthetic network and five real-world social networks, as shown in Table 1. PGP [49] is a social network formed by people that shares confidential information using the Pretty Good Privacy encryption algorithm. SynRand is a random generated synthetic network consisted by 10,000 nodes and 56,152 edges, of which the node degree obeys to Gaussian distribution. Other four social networks are selected from SNAP.[1] Fig. 2 shows the node degree distribution of each network.

The experiments are mainly divided into three separate phases. For the first phase, experiments on parameters setting of DBA are carried out to make sure the proposed DBA converges to global optimization effectively. Then comparisons between DBA and DPSO on *LIE* optimization are made in the six networks. In addition, a variation named DBA-np, in which the new bat is generated in the whole network topology by the random walk strategy, is derived. Comparisons of the effectiveness of the *CandidatesPool* strategy in DBA and DBA-np are carried out. For the third phase, four other state of the art algorithms, introduced in Section 2, are employed as baseline, experimental comparisons and statistical tests are simulated to illustrate the influence spread performance of the five algorithms.

- *CELF* (Cost-Effective Lazy Forward) [5] is a greedy-based algorithm with a "lazy forward" strategy by exploiting the submodularity property.
- *DPSO* (Discrete Particle Swarm Optimization) [20] is a meta-heuristic algorithm that selects seed nodes via the cooperative evolution of the particle swarm based on the local influence estimator model.
- *DDSE* (Degree-Descending Search Evolution) [38] is a novel evolutionary algorithm originated from Differential Evolutionary (DE) algorithm based on degree-descending search strategy for influence maximization.
- *SSA*: (Stop-and-Stare Algorithm) [34] is an optimal sampling framework based on reverse influence sampling in which the algorithm stops at exponential check points to verify if there is adequate statistical evidence on the solution quality.

All the procedures are coded by C++ language and executed on a PC platform with 4 Intel (R) Cores (TM), i7-4790 3.60 GHz CPU and 8 G memory.

5.2. Parameters setting

Experiments on the sensitivity of parameters setting including both γ and α to the convergence of DBA are conducted firstly. We assign the bat population size N and seed set size k the value of 30 and 50, respectively. For the random walk strategy in DBA, β is set to 3, μ and ν in Eq. (15) are set to 0.5, respectively. Secondly, experiments on *LIE* optimization and influence spread at propagation probability $p = 0.01$ under the IC model are simulated in the six networks. The maximal iterative generations for DBA, DPSO and DDSE are set to 100, respectively. For CELF, we run the Monte-Carlo simulation 10,000 times. According to the framework of SSA, parameters ϵ and δ are suggested to set $\epsilon < 1/4$ and $\delta = 1/n$, where n is the number of nodes. So ϵ and δ are set to 0.1 and 0.01 respectively in this paper. In addition, the simulation times for DBA, DPSO and DDSE are set to 1000 separately to calculate the average influence spread under the IC model.

5.2.1. Gamma setting

Experiments on *LIE* optimization are firstly carried out to select proper γ value so that DBA can converge to global optimal *LIE* effectively. As demonstrated in Section 4.1, the pulse emission rate r_i increases with bat i approaching to the prey and we have $\gamma > 0$. According to the condition constriction Eq. (7), weak ties exists between γ and α. Generally speaking, larger α value in Eq. (5) makes for extensive search in the solution space, so α is set to 0.9 initially to obtain applicable γ value.

As shown in Fig. 3, DBA achieves different *LIE* fitness value when γ varies from 0.1 to 0.9 at $\alpha = 0.9$ in the six networks. From the box plots shown in Fig. 3, we can see that DBA shows the best average performance at $\gamma = 0.3$ in most cases of the six networks. Specifically, DBA returns the best average *LIE* value at $\gamma = 0.3$ in cases of Fig. 3(B) and (D) though the best *LIE* tends to be achieved at $\gamma = 0.8$ and $\gamma = 0.5$, respectively. However, larger γ does not help to improve the *LIE* value in all the six scenarios. This can be illuminated by the updating rule of pulse rate according to Eq. (7). The pulse emission rate r_i converges rapidly to 1 with the increment of iteration index if we set the initial $r_i^0 = 1$ at a larger γ value, e.g. $\gamma = 0.9$, as shown in Fig. 4. Obviously, Larger γ value can accelerate the convergence of DBA, but it tends to lead the algorithm converges to local optimal solution easily. Conversely, smaller γ value will result in slower change in value of r_i, which plays a counterproductive role in associate with appropriate loudness A_i. Thus, γ is set to 0.3 to control the rhythm of pulse emission rate of each bat in the population for the following experiments.

5.2.2. Alpha setting

As demonstrated in Section 4.1, the loudness A_i of bat i in the population decreases with the bat approaching to the prey and we have $0 < \alpha < 1$. Intuitively, larger α provides more chances for DBA to explore extensively in the solution space, but the algorithm tends to converge slowly. In fact, to cooperate with γ and enable DBA to converge to global optimal solution, appropriate α setting plays an important role in the convergence of DBA. Therefore, experiments on α selection are carried out with α varying from 0.1 to 0.9 and $\gamma = 0.3$ in the six networks.

As shown in Fig. 5, we can see that DBA performs well at $\alpha = 0.3$ and achieves comparable *LIE* fitness value approximately to those scenarios at $\gamma = 0.3$, as shown in Fig. 3. One special case emerges in Fig. 5(C), where the best *LIE* is achieved by DBA at $\alpha = 0.2$ and the best average *LIE* is returned by DBA at $\alpha = 0.9$. It is noteworthy that the average *LIE* value at $\alpha = 0.3$ is always more stable than those returned

[1] http://snap.stanford.edu/data/index.html

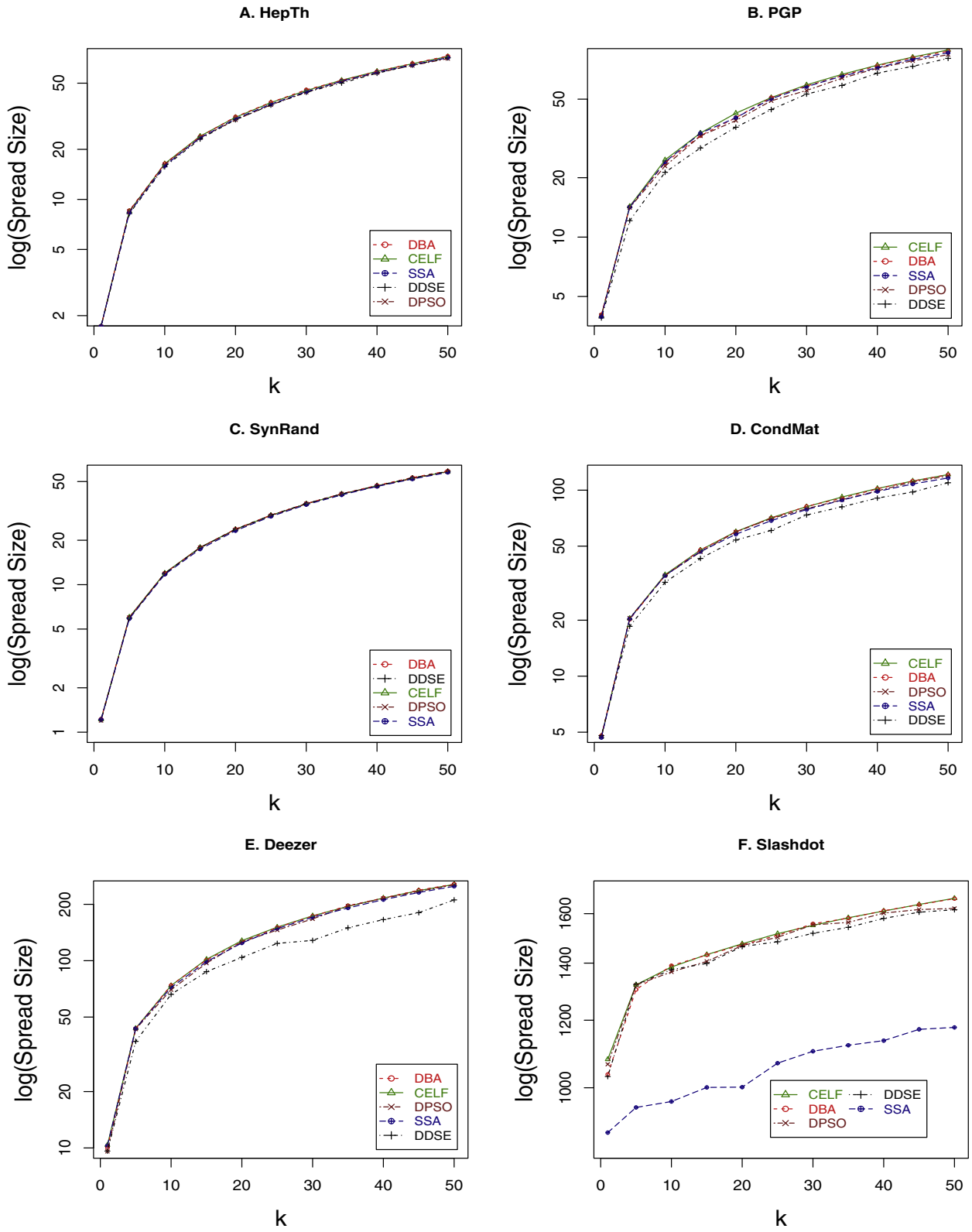

Fig. 8. Comparisons on influence spread under independent cascade model at propagation probability $p = 0.01$ of the five algorithms in the six networks.

by DBA at other scenarios, though the best *LIE* seems to be not achieved at $\alpha = 0.3$, such as in Fig. 5(B) and (D). Therefore, α and γ are set to 0.3 simultaneously for DBA to conduct subsequent experiments on influence maximization in social networks. As shown in Fig. 4, the loudness A_i has larger value range if α is set to 0.3 with the pulse emission rate approaching to the maximal r_i^0 when the γ is set to 0.3.

5.3. LIE optimization

Larger *LIE* fitness value implies higher quality of identification and wider influence spread of influential seed nodes. To show the performance of DBA in optimizing the *LIE* fitness, we conduct experiments on *LIE* optimization, meanwhile, the variation DBA-np and DPSO are

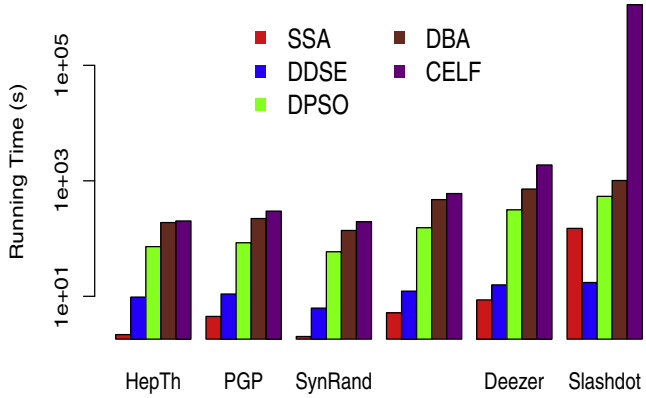

Fig. 9. Running time in log-scale form of the five algorithms on influence spread of targeted $k = 50$ seed nodes in the six networks.

employed to compare with DBA. Besides the same seed set size k, parameters including ω, c_1 and c_2 in DPSO are set to $\omega = 0.8$, $c_1 = c_2 = 2$, respectively, according to the statements in the original paper. As shown in Fig. 6, we can see firstly that all the three algorithms perform well on optimizing the *LIE* function, and they show similar performance when the seed size k is small. As shown in Fig. 6(A) and (E), the three algorithms achieve almost the same *LIE* value when the seed set size $k < 20$, as well as in Fig. 6(B), (D) and (F) when the seed set size $k \leq 10$. In Fig. 6(C), the three algorithms show similar performance in the synthetic random generated network, in which the node degree is normally distributed so that a Pareto optimal solution set is available to maximize the *LIE* estimator. In addition, the evolutionary curves simulated by DPSO are choppy as emerge in Fig. 6(D)–(F) in large-scale networks. However, DBA performs generally as the best one on optimizing the *LIE* and the evolutionary simulations are more smooth than those achieved by DBA-np and DPSO in the six networks at propagation probability $p = 0.01$ with the increase in seed size k.

All the observed results from the simulations prove that DBA is effective in optimizing *LIE* function. Comparisons between DBA and DBA-np show that the random walk strategy working on *CandidatesPool* provides more effective exploitation to the bat population and avoids DBA being trapped into local optimal location in the early stages. In addition, the experimental results also show that the local search strategy suggested in DPSO tends to expose the algorithm to be trapped into a suboptimal seed set easily in the early stage, because the semi-greedy strategy always fails to exploit more influential nodes based on

Table 2
Statistical test results of the multiple-problem Wilcoxons test for the five algorithms at $\alpha = 0.05$ and $\alpha = 0.1$ significance levels.

DBA vs	k	N+	N−	Z	p-value	Adjusted p-value		$\alpha = 0.1$	$\alpha = 0.05$
						Holm	Hochberg		
CELF	1	3	3	−0.524	0.600	1	0.6	No	No
	5	1	5	−1.782	0.075	0.3	0.3	No	No
	10	3	3	−0.105	0.917	1	0.917	No	No
	15	1	5	−1.782	0.075	0.15	0.15	No	No
	20	2	4	−1.363	0.173	0.173	0.173	No	No
	25	2	4	−1.153	0.249	0.249	0.249	No	No
	30	4	2	−0.314	0.753	0.753	0.753	No	No
	35	2	4	−0.734	0.463	0.463	0.463	No	No
	40	3	3	−0.105	0.971	0.971	0.971	No	No
	45	3	3	−0.105	0.917	0.971	0.971	No	No
	50	3	3	−0.314	0.753	0.753	0.753	No	No
DPSO	1	5	1	−0.943	0.345	1	0.6	No	No
	5	3	3	−0.314	0.753	0.753	0.753	No	No
	10	6	0	−2.201	0.028	1	0.971	No	No
	15	6	0	−2.201	0.028	0.112	0.084	No	No
	20	6	0	−2.201	0.028	0.112	0.056	No	No
	25	6	0	−2.201	0.028	0.112	0.056	No	No
	30	6	0	−2.201	0.028	0.112	0.056	No	No
	35	6	0	−2.201	0.028	0.112	0.056	No	No
	40	6	0	−2.201	0.028	0.112	0.056	No	No
	45	6	0	−2.201	0.028	0.112	0.056	No	No
	50	6	0	−2.201	0.028	0.112	0.056	No	No
SSA	1	4	2	−0.943	0.345	1	0.6	No	No
	5	4	2	−1.153	0.249	0.747	0.69	No	No
	10	5	1	−1.782	0.075	0.225	0.225	No	No
	15	5	1	−1.363	0.173	0.173	0.173	No	No
	20	6	0	−2.201	0.028	0.112	0.056	No	No
	25	6	0	−2.201	0.028	0.112	0.056	No	No
	30	6	0	−2.201	0.028	0.112	0.056	No	No
	35	6	0	−2.201	0.028	0.112	0.056	No	No
	40	6	0	−2.201	0.028	0.112	0.056	No	No
	45	6	0	−2.201	0.028	0.112	0.056	No	No
	50	6	0	−2.201	0.028	0.112	0.056	No	No
DDSE	1	5	1	−1.782	0.075	0.3	0.3	No	No
	5	5	1	−0.943	0.345	0.747	0.69	No	No
	10	6	0	−2.201	0.028	0.112	0.112	No	No
	15	6	0	−2.201	0.028	0.112	0.084	No	No
	20	6	0	−2.201	0.028	0.112	0.056	No	No
	25	6	0	−2.201	0.028	0.112	0.056	No	No
	30	6	0	−2.201	0.028	0.112	0.056	No	No
	35	6	0	−2.201	0.028	0.112	0.056	No	No
	40	6	0	−2.201	0.028	0.112	0.056	No	No
	45	6	0	−2.201	0.028	0.112	0.056	No	No
	50	6	0	−2.201	0.028	0.112	0.056	No	No

topology-specially one-hop area. Thus, we adopt DBA to conduct influence spread experiments under the IC model and validate its effectiveness by comparing it with other four algorithms.

Fig. 7 shows the running time of the three algorithms in optimizing the *LIE* function. The running time of each algorithm shown in Fig. 7 is self evident to select the targeted $k = 50$ seed nodes. All the three algorithms achieve the targeted seed nodes rapidly, but the time difference is still distinct. Generally, both DBA and DBA-np require more computational time than DPSO to get the optimal seed set, but they are more effective than DPSO in identifying the optimal seed set, as we can see in Fig. 6. The results in Fig. 7 also indicate that DBA outperforms DBA-np on running time about 16.5% on average, which is benefited from the *CandidatesPool* generated for random walk strategy.

5.4. Comparisons of typical algorithms

To show the performance of DBA on influence spread, we choose four other state of the art algorithms discussed in Section 5.1 as baseline. The evolutionary simulations shown in Fig. 8 under IC model illustrate the influence spread of the five algorithms at the given propagation probability $p = 0.01$. As shown in Fig. 8, all the five algorithms perform steadily on influence spread at the propagation probability $p = 0.01$ in the six large networks with different node degree distribution. As illustrated in Fig. 8(A), (C) and (E), DBA performs as the best one compared with other four algorithms. In Fig. 8(B), (D) and (F), DBA achieves comparable influence spread to CELF, and shows better performance than SSA, DPSO and DDSE. Besides the similar updating rules for the velocity and location of DBA and DPSO, the probabilistic greedy-based local search strategy and the random walk for DBA play an important role in prompting the algorithm to explore global optimal seed set. Benefiting from the local search strategy, each bat in the population can be updated by potential more influential candidate nodes from the one-hop area of the temporary seed nodes, and the *CandidatesPool* can avoid the algorithm searching blindly in the whole network space effectively. However, the local search strategy in the framework of DPSO tends to fail in exploiting local influential nodes to replace less influential nodes in the seed set, which would lead the algorithm to local optimal solution easily.

In Fig. 8(F), benefiting from the property of “submodularity”, CELF tends to select the influential node that can bring the most marginal gain and performs satisfying influence spread in the six networks. With the increase in node diversity, the performance of the five algorithms is obviously different, DDSE and SSA tend to suffer from low solution accuracy easily especially in Fig. 8(D), (E) and (F). According to the influence spread results, other algorithms achieve comparable results compared with CELF when the seed size k is small, such as $k < 10$ in Fig. 8(B) and (D), but the influence spread is distinct especially when $k > 15$ as shown in Fig. 8(D), (E) and (F). As shown in Fig. 9, DDSE evolutes faster than both DPSO and DBA, but the influence estimation mechanism used in the algorithm always fails to identify influential seed nodes effectively so that it always acts as the worst method compared with other four algorithms. Based on the random reverse influence sampling mechanism, SSA runs as the fastest algorithm in the six networks at propagation probability $p = 0.01$, but it cannot provide high identification guarantee compared with CELF and DBA, especially in Fig. 8(F). As shown in Fig. 8(F), CELF can achieve the guarantee solution within bearable time budget at the small propagation probability $p = 0.01$ under the IC model, but it tends to need unbearable time to identify the targeted seed set when the propagation probability is large, such as $p = 0.05$. As far as the time computation concerned, the experimental simulations prove DBA to be a promising algorithm in identifying influential nodes effectively for influence maximization in large-scale networks.

5.5. Statistical tests

To verify the effectiveness of DBA independently, we also conduct rigorous statistical tests in terms of quartile statistics to check whether there is a high level of statistical significance in the results of the five algorithms for the influence maximization problem in the six networks. In each of the network, 11 k scenarios ($k = 1, 5, 10, ..., 50$) are considered as independent problems, moreover, the parameter-free hypothesis tests on each k scenario in the six networks are carried out separately. The multiple-problem Wilcoxon tests [50] are performed to check the behaviors of the five algorithms, in which Holm procedure and Hochberg procedure are used as post-hoc procedures. It is necessary to emphasize that multiple-problem Wilcoxons test and post-hoc procedures are accomplished by using the SPSS software in this paper.

Table 2 summarizes the statistical analysis results by using DBA as the baseline. According to the statistical results, we can see that DBA is significantly better than DDSE and SSA, meanwhile, it achieves similar performance compared to the greedy-based CELF algorithm.

6. Conclusion

Influence maximization problem remains as an open research topic of viral marketing and social network analysis, etc. It is necessary to further develop effective and efficient algorithms that are scalable to the problem in large-scale networks. In this paper, a discrete bat algorithm based on network topology is proposed for the influence maximization in large-scale networks. According to the framework of discrete bat algorithm, a probabilistic greedy-based local search based on network topology is presented to avoid DBA being trapped into suboptimal solution easily. A random walk strategy according to the original ideology of BA is redesigned to enhance the exploitation operation in the solution space. Meanwhile, A *CandidatesPool* maintaining potential influential nodes is generated according to the contribution of each node’s degree centrality and its closeness centrality to the network topology for the random walk strategy to accelerate DBA converges effectively. The experimental results show that DBA achieves better *LIE* fitness in different scale of networks compared with DPSO, and the results based on IC model at propagation probabilities $p = 0.01$ validate DBA to be a promising effective and robust algorithm with low time complexity compared with CELF for influence maximization problem.

Although the probabilistic greedy-based local search strategy is helpful to improve the identification of seed nodes, DBA tends to suffer from the problem of high computational time due to the greedy-based mechanism. Therefore, developing more effective influence estimator of given node set and efficient metaheuristic evolutionary rules based on collective intelligence is the mainly focus of our future work.

Acknowledgment

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

References

[1] R.M. Bond, C.J. Fariss, J.J. Jones, A.D. Kramer, C. Marlow, J.E. Settle, J.H. Fowler, A 61-million-person experiment in social influence and political mobilization. Nature 489 (7415) (2012) 295–298.

[2] P. Domingos, M. Richardson, Mining the network value of customers, Proceedings of the 2001 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2001), pp. 57–66.

[3] J. Goldenberg, B. Libai, Using complex systems analysis to advance marketing theory development: modeling heterogeneity effects on new product growth through stochastic cellular automata, academy of marketing science review 9, Mon. Labor Rev. 31 (3) (2001) 8–11.

[4] Y. Cho, J. Hwang, D. Lee, Identification of effective opinion leaders in the diffusion

of technological innovation: a social network approach, Technol Forecast Soc Change 79 (1) (2012) 97–106.
[5] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen, N. Glance, Cost-effective outbreak detection in networks, Proceedings of the 2007 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2007), pp. 420–429.
[6] J. Borge-Holthoefer, Y. Moreno, Absence of influential spreaders in rumor dynamics, Phys. Rev. E Stat. Nonlinear Soft Matter Phys. 85 (2 Pt 2) (2012) 026116.
[7] Y. Li, S. Ma, Y. Zhang, R. Huang, Kinshuk, An improved mix framework for opinion leader identification in online learning communities, Knowl. Based Syst. 43 (2) (2013) 43–51.
[8] W. Chen, T. Lin, C. Yang, Efficient Topic-aware Influence Maximization Using Preprocessing, CoRR (2014). abs/1403.0057
[9] G. Li, S. Chen, J. Feng, W.S. Li, W.S. Li, Efficient location-aware influence maximization, Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, (2014), pp. 87–98.
[10] X. Li, X. Cheng, S. Su, C. Sun, Community-based seeds selection algorithm for location aware influence maximization, Neurocomputing 275 (2017) 1601–1613.
[11] L. Guo, D. Zhang, W. Wu, G. Cong, K.L. Tan, Influence maximization in trajectory databases, IEEE Trans. Knowl. Data Eng. 29 (3) (2017) 627–641.
[12] H. Li, L. Pan, P. Wu, Dominated competitive influence maximization with time-critical and time-delayed diffusion in social networks, J. Comput. Sci. (2017), https://doi.org/10.1016/j.jocs.2017.10.015. (in press)
[13] F. Morone, H.A. Makse, Influence maximization in complex networks through optimal percolation, Nature 527 (7579) (2015) 544.
[14] M.M.D. Khomami, A. Rezvanian, N. Bagherpour, M.R. Meybodi, Minimum positive influence dominating set and its application in influence maximization: a learning automata approach, Appl. Intell. 48 (3) (2017) 570–593.
[15] D. Kempe, J. Kleinberg, Maximizing the spread of influence through a social network, Proceedings of the 2003 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2003), pp. 137–146.
[16] A. Goyal, W. Lu, L.V.S. Lakshmanan, Celf++: optimizing the greedy algorithm for influence maximization in social networks, Proceedings of the 2011 International Conference Companion on World Wide Web, (2011), pp. 47–48.
[17] F. Zhou, J. Jiao, B. Lei, A linear threshold-hurdle model for product adoption prediction incorporating social network effects, Inf. Sci. (NY) 307 (20) (2015) 95–109.
[18] Y. Tang, Y. Shi, X. Xiao, Influence maximization in near-linear time: a martingale approach, Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, (2015), pp. 1539–1554.
[19] W. Chen, Y. Wang, S. Yang, Efficient Influence Maximization in Social Networks, Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'2009) (2009) 199–207, https://doi.org/10.1145/1557019.1557047.
[20] M. Gong, J. Yan, B. Shen, L. Ma, Q. Cai, Influence maximization in social networks based on discrete particle swarm optimization, Inf. Sci. (NY) 367 (2016) 600–614.
[21] T. Zhu, B. Wang, B. Wu, C. Zhu, Maximizing the spread of influence ranking in social networks, Inf. Sci. (NY) 278 (2014) 535–544.
[22] J.-R. Lee, C.-W Chung, A query approach for influence maximization on specific users in social networks, IEEE Trans. Knowl. Data Eng. 27 (2) (2015) 340–353.
[23] Y. Wang, G. Cong, G. Song, K. Xie, Community-based greedy algorithm for mining top-k influential nodes in mobile social networks, Proceedings of the 2010 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2010), pp. 1039–1048.
[24] X. Zhang, J. Zhu, Q. Wang, H. Zhao, Identifying influential nodes in complex networks with community structure, Knowl. Based Syst. 42 (2) (2013) 74–84.
[25] J. Shang, S. Zhou, X. Li, L. Liu, H. Wu, CoFIM: a community-based framework for influence maximization on large-scale networks, Knowl. Based Syst. 117 (2016) 88–100.
[26] W.X. Lu, C. Zhou, J. Wu, Big social network influence maximization via recursively estimating influence spread, Knowl. Based Syst. 113 (2016) 143–154.
[27] M. Han, M. Yan, Z. Cai, Y. Li, X. Cai, J. Yu, Influence maximization by probing partial communities in dynamic online social networks, Trans. Emerg. Telecommun. Technol. 28 (4) (2017) e3054, https://doi.org/10.1002/ett.3054.
[28] Q. Zhao, H. Lu, Z. Gan, X. Ma, A k-shell decomposition based algorithm for influence maximization, in: P. Cimiano, F. Frasincar, G.-J. Houben, D. Schwabe (Eds.), Proceedings of the 2015 Engineering the Web in the Big Data Era, Springer International Publishing, Cham, 2015, pp. 269–283.
[29] S. Yeruva, T. Devi, Y.S. Reddy, Selection of influential spreaders in complex networks using Pareto shell decomposition, Phys. A Stat. Mech. Appl. 452 (2016) 133–144.
[30] S. Kim, D. Kim, J. Oh, J.H. Hwang, W.S. Han, W. Chen, H. Yu, Scalable and parallelizable influence maximization with random walk ranking and rank merge pruning, Inf. Sci. (NY) 45 (2017) 171–189.
[31] C. Borgs, M. Brautbar, J. Chayes, B. Lucier, Maximizing social influence in nearly optimal time, Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms (SODA'14), Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2014, pp. 946–957.
[32] Y. Tang, X. Xiao, Y. Shi, Influence maximization: near-optimal time complexity meets practical efficiency, Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD '14, (2014), pp. 75–86.
[33] E. Cohen, D. Delling, T. Pajor, R.F. Werneck, Sketch-based Influence Maximization and Computation: Scaling up with Guarantees, Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM'14), ACM, New York, NY, USA, 2014, pp. 629–638.
[34] H.T. Nguyen, M.T. Thai, T.N. Dinh, Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks, Proceedings of the 2016 International Conference on Management of Data, SIGMOD '16, ACM, New York, NY, USA, 2016, pp. 695–710.
[35] Q. Jiang, G. Song, G. Cong, Y. Wang, W. Si, K. Xie, Simulated annealing based influence maximization in social networks, Proceedings of the 2011 AAAI Conference on Artificial Intelligence, (2011), pp. 127–132.
[36] C.P. Sankar, S. Asharaf, K.S. Kumar, Learning from bees: an approach for influence maximization on viral campaigns, PLoS One 11 (12) (2016) e0168125.
[37] M. Gong, C. Song, C. Duan, L. Ma, An efficient memetic algorithm for influence maximization in social networks, IEEE Comput. Intell. Mag. 11 (3) (2016) 22–33.
[38] L. Cui, H. Hu, S. Yu, Q. Yan, Z. Ming, Z. Wen, N. Lu, DDSE: a novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks, J. Netw. Comput. Appl. 103 (2018) 119–130.
[39] X.S. Yang, A new metaheuristic bat-inspired algorithm, Comput. Knowl. Technol. 284 (2010) 65–74.
[40] X. Yang, Bat algorithm for multi-objective optimisation, Int. J. Bio-Inspired Comput. 3 (5) (2011) 267–274.
[41] A.H. Gandomi, X.S. Yang, A.H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Comput. Appl. 22 (6) (2013) 1239–1255.
[42] R. Soto, B. Crawford, R. Olivares, S. Niklander, F. Johnson, F. Paredes, E. Olgun, Online control of enumeration strategies via bat algorithm and black hole optimization, Nat. Comput. 16 (2) (2017) 241–257.
[43] Y. Saji, M.E. Riffi, A novel discrete bat algorithm for solving the travelling salesman problem, Neural Comput. Appl. 27 (7) (2015) 1–14.
[44] E. Osaba, X.S. Yang, F. Diaz, P. Lopez-Garcia, R. Carballedo, An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems, Eng. Appl. Artif. Intell. 48 (C) (2016) 59–71.
[45] S.C. Satapathy, N.S.M. Raja, V. Rajinikanth, A.S. Ashour, N. Dey, Multi-level image thresholding using Otsu and chaotic bat algorithm, Neural Comput. Appl. 29 (12) (2018) 1285–1307.
[46] H. Liang, Y. Liu, F. Li, Y. Shen, A multiobjective hybrid bat algorithm for combined economic/emission dispatch, Int. J. Electr. Power Energy Syst. 101 (2018) 103–115.
[47] N.A. Christakis, J.H. Fowler, Connected: The Surprising Power of our Social Networks and How They Shape our Lives – How Your Friends' Friends' Friends Affect Everything you Feel, Think, and Do, Little, Brown, 2011.
[48] S. Pei, L. Muchnik, J.S.A.Jr Andrade, Z. Zheng, H.A. Makse, Searching for superspreaders of information in real-world social media, Sci. Rep. 4 (2014) 5547.
[49] S. Gregory, Finding Overlapping Communities Using Disjoint Community Detection Algorithms, Springer Berlin Heidelberg, 2009.
[50] S. Garca, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms behaviour: a case study on the CEC 2005 special session on real parameter optimization, J. Heurist. 15 (6) (2009) 617–644.