

Identifying influential spreaders in social networks: A two-stage quantum-behaved particle swarm optimization with Lévy flight

Pengli Lu(卢鹏丽), Jimao Lan(揽继茂), Jianxin Tang(唐建新), Li Zhang(张莉), Shihui Song(宋仕辉), and Hongyu Zhu(朱虹羽)

Citation: Chin. Phys. B, 2024, 33 (**1**): 018901. DOI: 10.1088/1674-1056/acd3e0
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Topological phase transition in network spreading

Fuzhong Nian(年福忠) and Xia Zhang(张霞)
Chin. Phys. B, 2023, 32 (**3**): 038901. DOI: 10.1088/1674-1056/ac7bfa

A novel similarity measure for mining missing links in long-path networks

Yijun Ran(冉义军), Tianyu Liu(刘天宇), Tao Jia(贾韬), and Xiao-Ke Xu(许小可)
Chin. Phys. B, 2022, 31 (**6**): 068902. DOI: 10.1088/1674-1056/ac4483

Asynchronism of the spreading dynamics underlying the bursty pattern

Tong Wang(王童), Ming-Yang Zhou(周明洋), Zhong-Qian Fu(付忠谦)
Chin. Phys. B, 2020, 29 (**5**): 058901. DOI: 10.1088/1674-1056/ab81fe

Identifying influential spreaders in complex networks based on entropy weight method and gravity law

Xiao-Li Yan(闫小丽), Ya-Peng Cui(崔亚鹏), Shun-Jiang Ni(倪顺江)
Chin. Phys. B, 2020, 29 (**4**): 048902. DOI: 10.1088/1674-1056/ab77fe

Uncovering offline event similarity of online friends by constructing null models

Wenkuo Cui(崔文阔), Jing Xiao(肖婧), Ting Li(李婷), Xiaoke Xu(许小可)
Chin. Phys. B, 2019, 28 (**6**): 068901. DOI: 10.1088/1674-1056/28/6/068901

Identifying influential spreaders in social networks: A two-stage quantum-behaved particle swarm optimization with Lévy flight

Pengli Lu(卢鹏丽)[2], Jimao Lan(揽继茂)[2], Jianxin Tang(唐建新)[1,2,†], Li Zhang(张莉)[2], Shihui Song(宋仕辉)[2], and Hongyu Zhu(朱虹羽)[2]

[1] *Wenzhou Engineering Institute of Pump&Valve, Lanzhou University of Technology, Wenzhou* 325100, *China*
[2] *School of Computer and Communication, Lanzhou University of Technology, Lanzhou* 730050, *China*

(Received 9 February 2023; revised manuscript received 8 May 2023; accepted manuscript online 10 May 2023)

The influence maximization problem aims to select a small set of influential nodes, termed a seed set, to maximize their influence coverage in social networks. Although the methods that are based on a greedy strategy can obtain good accuracy, they come at the cost of enormous computational time, and are therefore not applicable to practical scenarios in large-scale networks. In addition, the centrality heuristic algorithms that are based on network topology can be completed in relatively less time. However, they tend to fail to achieve satisfactory results because of drawbacks such as overlapped influence spread. In this work, we propose a discrete two-stage metaheuristic optimization combining quantum-behaved particle swarm optimization with Lévy flight to identify a set of the most influential spreaders. According to the framework, first, the particles in the population are tasked to conduct an exploration in the global solution space to eventually converge to an acceptable solution through the crossover and replacement operations. Second, the Lévy flight mechanism is used to perform a wandering walk on the optimal candidate solution in the population to exploit the potentially unidentified influential nodes in the network. Experiments on six real-world social networks show that the proposed algorithm achieves more satisfactory results when compared to other well-known algorithms.

Keywords: social networks, influence maximization, metaheuristic optimization, quantum-behaved particle swarm optimization, Lévy flight

PACS: 89.75.–k, 01.75.+m, 03.67.Ac, 05.40.Fb **DOI:** 10.1088/1674-1056/acd3e0

1. Introduction

As the pandemic spread around the world, lifestyles and the ways in which people obtain information were unexpectedly transferred to online mediums.[1] The virtual world has provided essential services, while reducing the distance between individuals. In particular, online social platforms, such as Facebook, WeChat, and TikTok, have promoted the growth of information consumption with the growing number of users and have nourished the emergence of online viral marketing on social networks.

While online social networks provide convenience to the users, they also offer opportunities to companies to market their products.[3] Generally, companies first select a group of influential individuals as adopters and give them free access to the product in the expectation that they will recommend the product to their family and friends through word of mouth. If there are strong connections among these users to sustain the exponential spread of the message, then the viral marketing campaign will probably be successful and the referral behavior will, partly, influence the final choice of the purchasers. This scenario has applications in practical campaigns, such as rumor restriction,[4] where one would identify the superspreaders of rumors and restrict them to reduce the risk to society.

Due to the inherent heterogeneity of individuals and the different ways of communication, the influence maximization (IM) problem here refers to the selection of a set of independent individuals that can maximize the coverage of information dissemination through the social relationships among the individuals.

The IM problem was first proposed by Domingos and Richardson,[5] who suggested that we should consider the consumer market as a social network and model it as a Markov random field. Researchers have since proposed a series of IM algorithms to conduct an in-depth study. Kempe *et al.*[6] proved that the optimization problem of selecting the most influential nodes is NP-hard under the independent cascade (IC) model and linear threshold (LT) model, and proposed a natural greedy strategy, promising that the obtained solution is approximately within 63% of the optimal based on submodular function analysis. However, to accurately evaluate the marginal gain of influence of each candidate node, thousands of Monte Carlo simulations are required in each round of seed selection. This leads to the high time complexity $O(knmR)$ of the algorithm. In particular, the running time of the algorithm increases exponentially with the increase of the network scale, thereby limiting its application in large-scale social networks.

Researchers have subsequently tried to directly select k nodes as influential nodes according to the network topology features, such as degree centrality,[7] PageRank,[8] and so on.

†Corresponding author. E-mail: tangjx@lut.edu.cn

© 2024 Chinese Physical Society and IOP Publishing Ltd

Alshahrani *et al.*[9] tried to combine the local influence with global influence of each node by using classical centrality metrics and proposed two algorithms, which are called MaxCDegKatz d-hops and MinCDegKatz d-hops. However, the approach of using network topology centrality fails to provide stable performance guarantees due to the lack of diversity in the selected nodes in different networks. In addition, community-based approaches have been developed to find influential nodes in the network. Bozorgi *et al.*[10] considered the influence of nodes within communities as a local influence and the influence of communities in the whole network as a global influence, and proposed INICIM to evaluate influential nodes by combining global influence and local influence. Cai *et al.*[11] explored the community property to improve the efficiency of the algorithm and proposed a community-based greedy algorithm to identify the seed nodes. The advantage of community-based methods is that they provide a good trade-off between influence propagation and runtime. However, there are two problems: first, there is a lack of effective strategies to reduce the search space when selecting select seed nodes in large networks; and second, some of the algorithms suffer from the community overlap problem.

In recent years, metaheuristic algorithms have been proposed to efficiently solve the IM problem. Zareie *et al.*[12] defined the cost function as the influence of nodes and the distance between them, and proposed a population-based gray wolf algorithm to solve the IM problem. Li *et al.*[13] proposed a discrete crow optimization metaheuristic algorithm to efficiently solve the IM problem. This type of algorithm has a significant improvement in running time when compared with the greedy strategy-based approach. There is also a substantial improvement in the solution accuracy when compared to the centrality-based methods. However, the existing metaheuristic algorithms have some limitations because they may fall into another local optimum when they try to get rid of the local optimum. With this in mind, we develop a discrete two-stage metaheuristic optimization (DTMO) algorithm combining QPSO and Lévy flight to efficiently identify the influential nodes in the network. The main contributions of this paper are as follows:

A new two-stage metaheuristic optimization algorithm framework was formulated to solve the IM problem.

(i) The evolution rules of the quantum-behaved particle swarm optimization algorithm were modified and a discrete quantum-behaved particle swarm optimization (DQPSO) algorithm with nonlinearly decreasing randomized crossover operation was developed.

(ii) A discrete Lévy flight (DLF) algorithm with automatic candidate pool selection based on greedy strategy was presented to enhance the performance of the DQPSO.

(iii) A new method for population diversity calculation was designed and a novel algorithmic transformation strategy was introduced using this method.

(iv) Our method was compared with other well-known methods on six real-world social networks. The experiments showed that our method can obtain comparable effects to the algorithms based on greedy strategy but with low time complexity.

The rest of this paper is organized as follows. Section 2 reviews and discusses the related work. The problem description, the fitness assessment function, and the propagation model are given in Section 3. Section 4 presents the original QPSO and Lévy algorithms and describes our proposed framework in detail. The experimental results and analysis are given in Section 5. Finally, this study is summarized and future research directions are proposed in Section 6.

2. Related work

The IM problem was first proposed from a network perspective by Domingos and Richardson,[5] where the authors argued that consumers and the individuals around them are interconnected and have an interacting influence, which was modeled through Markov random field theory. Kempe *et al.*[6] transformed the IM problem into a combinatorial optimization problem. They adopted the Monte Carlo simulation mechanism to evaluate the influence of candidate nodes and introduced a greedy algorithm that was based on a hill-climbing search strategy to find the influential nodes. To reduce the expensive computational cost that arose from the Monte Carlo simulations, Leskovec *et al.*[14] developed an improved greedy algorithm that was scalable to large-scale social networks. Their experiments showed that it is 700 times faster than the simple greedy algorithm, while the results are nearly optimal. Subsequently, Goyal *et al.*[15] proposed a superior version called CELF++, which is 35%–55% faster than CELF. Following this seminal work, Zhang *et al.*[16] introduced a residual-based algorithm RCELF that can achieve good time efficiency, low memory consumption, and approximate quality of the results. Compared with the original greedy algorithm, the improved greedy algorithm has a certain degree of improvement in terms of time cost. However, some algorithms need to record the states during propagation to reduce the number of Monte Carlo simulations. Meanwhile, some algorithms increase the memory consumption, which means that they cannot be scaled to real large-scale social networks.

Several heuristic algorithms have been put forward to tackle IM variations. Chen *et al.*[17] gave a more accurate discount value to the neighbors of the nodes selected as seeds and proposed the DegreeDiscount under the IC model with small propagation probability. Their experiments showed that the fine-tuned heuristic can provide a truly scalable solution to the IM problem with satisfactory propagation range and greater efficiency. Zareie *et al.*[18] proposed a special hierarchical

measure to provide sufficient information about the topological position of the nodes, which ranks the influence of nodes more accurately than other state-of-the-art measures. Wang *et al.*[19] introduced a new metric, which they called node key degree, to measure the importance of nodes, and proposed the ITÖ algorithm to balance the conflict between exploration and exploitation. More recently, considering the dynamic nature and local aggregation factors on diffusion, Li *et al.*[20] adopted various entropy calculations to obtain the cohesion between neighboring nodes and then identify whether the node has the ability to become a propagable pioneer of other nodes. Meng *et al.*[21] combined H-index, K-shell iteration factor, and clustering coefficient to attach weights to connected edges, which were in turn combined with the neighborhood, position and topology of nodes in the network to identify influential nodes. Li *et al.*[22] combined an improved gravity model with the community detection method to identify influence propagators in the network by finding bridge nodes in the network's topology. Compared with traditional centrality methods, the methods combining the node characteristics can achieve a certain degree of improvement in accuracy. However, there is also a high time complexity, which makes it difficult to apply these methods to large-scale networks. In addition, some methods only measure the local structure of the network without diversity, and therefore lack solution stability when dealing with the IM problem.

Some recent research interest has focused on the use of metaheuristic algorithms, broadly referring to the construction of low computational cost influence evaluating models and the utilization of appropriate evolutionary optimization strategies to view the IM problem as a fitness optimization problem. Jiang *et al.*[23] were the first to use the fitness evaluation function, named expected diffusion value (EDV), as an influence evaluation measurement and proposed a simulated annealing algorithm to optimize EDV to identify a set of influential seed nodes. Their experimental results showed that the proposed algorithm runs 2–3 orders of magnitude faster than the state-of-the-art greedy algorithm, while improving the optimal solution accuracy. Gong *et al.*[24] proposed a discrete particle swarm optimization algorithm to map candidate seeds to particles in the population and established a novel evaluation function called local influence estimation (LIE), which can be evaluated more accurately under the IC model with lower time complexity. A metaheuristic discrete bat algorithm based on the collective intelligence of individual bats from the population was proposed by Tang *et al.*[25] This algorithm combines the evolutionary rules of the original bat algorithm and designs a seed node candidate pool to enhance the search capability of the algorithm, and finally achieves satisfactory experimental results. By analyzing the efficiency of the greedy algorithm, Cui *et al.*[26] proposed a degree-descending search strategy and developed a more efficient evolutionary algorithm, named degree-descending search evolution (DDSE), through the operations of mutation, crossover, and greedy selection. Singh *et al.*[27] extended EDV to a two-hop area to realize a more accurate evaluation of seed nodes and proposed a variant of discrete particle swarm optimization (DPSO) based on the mechanism of learning automata. To maximize the distance between the seed nodes and ensure that different parts of the network are reached, Zareie *et al.*[12] solved the IM problem by optimizing the influence of nodes and the distance between them via the gray wolf optimization algorithm, which performed well in experimental results and had lower computational cost. Wang *et al.*[28] developed an influence evaluation model based on the total valuation and valuation differences of neighboring nodes. The authors developed an evolutionary strategy with local crossover and variation on natural moth evolutionary rules. Their experiments showed that the method is effective and robust in dealing with the IM problem. After conducting extensive experiments, Weskida *et al.*[29] showed computationally that evolutionary algorithms not only overcome the limitations of greedy algorithms but also have several advantages, such as the transferability of their parameters. However, a reasonable discrete evolutionary mechanism can provide a balanced trade-off in terms of solution accuracy, time consumption, and even memory management. Therefore, the design of more effective influence evaluation mechanisms and more reasonable evolution mechanisms deserves further discussion.

Quantum-behaved particle swarm optimization, as a swarm intelligence algorithm, was proposed by Sun *et al.*[30] from a quantum mechanical perspective by combining some features of the original particle swarm optimization (PSO) algorithm. Due to its efficiency and robustness, this algorithm has recently been applied to solve problems in various fields. For example, a hybrid inversion method based on the quantum particle swarm optimization method was introduced to solve the electromagnetic inverse problem by Yang *et al.*[31] Bajaj *et al.*[32] proposed a discrete quantitative particle swarm optimization to improve the efficiency of test case prioritization. There are many other studies and applications of QPSO, such as path planning and the design of mobile robots in the workspace,[33] the constrained portfolio selection problem,[34] and the network clustering problem.[35] The results of these studies show that the QPSO algorithm provides strong robustness and solution efficiency thanks to its excellent evolutionary rules.

QPSO has attracted the attention of many researchers due to the simplicity of the evolution equation, few control parameters, and fast convergence, but it also suffers from problems such as premature convergence. As a random wandering strategy, Lévy flight is mainly adopted by combinatorial optimization algorithms to effectively solve intractable optimization problems, such as the green scheduling problem of the flex-

ible manufacturing cell.[36] However, the use of combinatorial algorithms based on QPSO and Lévy flight for the IM problem has not yet been reported in the literature. Therefore, exploring reasonable evolutionary mechanisms based on QPSO and Lévy flight to effectively identify influential nodes is worth further investigation.

3. Preliminary information

3.1. Influence maximization

In the study of IM, a social network can be abstracted as a graph $G=(V,E)$, where $V=\{v_1,v_2,\ldots,v_n\}$ is the set of vertices, representing the individuals or organizations in the social network, and $E=\{e_1,e_2,\ldots,e_m\}$ is the set of connected edges, indicating the existence of connections and cooperation between individuals or organizations in the social network. Note that n and m denote the number of vertices and the number of connected edges in the network, respectively.

For a given graph G, a set of k ($k \ll n$) influential nodes are selected and ignited. The number of nodes activated by the seed set in the graph is expected to be maximized under a specific propagation model. This problem can be formalized as

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

where $\sigma(\cdot)$ is a measure of influence spread, S represents a candidate seed set with k nodes and S^* denotes the optimal seed set that maximizes the influence spread.

3.2. Influence estimation function

In the IM problem, the influence evaluation methods are divided into two main categories: (i) in the first category, a Monte Carlo simulation evaluates the propagation outcomes of the influential nodes; and (ii) in the second category, information based on the neighborhood structure characteristics of the nodes is utilized to estimate the diffusion effect of the seed nodes.

Although the Monte Carlo simulation method can achieve high accuracy, it is not well satisfied with practical scenarios due to its tremendous computational complexity. Therefore, to reduce computational cost, Jiang *et al.*[23] proposed an influence evaluation function based on the direct adjacent neighbors of its corresponding seed nodes. Inspired by the principle of two-degree theory on influence spreading,[37] Gong *et al.*[24] proposed a LIE function, which approximates the expected influence spread based on the two-hop neighbors area of the influential nodes,

$$\mathrm{LIE}(S) = k + \left(1 + \frac{\sum\limits_{u\in N_S^{(2)}\setminus S} p_u^* d_u^*}{|N_S^{(1)}\setminus S|}\right)\sigma_1^*(S), \tag{2}$$

$$\sigma_1^*(S) = \sum_{i\in N_S^{(1)}\setminus S}\left(1-\prod_{\substack{(i,j)\in E\\ j\in S}}(1-p_{i,j})\right), \tag{3}$$

where $|N_S^{(1)}\in S|$ and $|N_S^{(2)}\in S|$ are the one-hop and two-hop area of the seed set S, respectively. p_u^* is the constant active probability of a diffusion model, d_u^* is the number of edges of node u within $N_S^{(1)}$ and $N_S^{(2)}$, and $\sigma_1^*(S)$ represents the expected influence spread of one-hop area of the seed set S. Obviously, the IM problem is transformed into the problem of maximizing the LIE function. Subsequently, LIE will be employed as the influence estimation function of the candidate nodes.

3.3. Influence propagation model

Currently, there are three widely spreading models for simulating influence propagation in the IM problem: the IC model, the LT model, and the weighted cascade model. Based on the influence estimator, we employ the classical IC model to simulate the spread of influence in given networks. In the IC model, nodes have two states: active and inactive. The node can only be converted from the inactive state to the active state during the propagation process, and vice versa. In the diffusion process, when a node is activated at time t, it has a single chance at time $t+1$ to activate its direct inactive neighbors with probability p. When a new node is activated, it repeats the previous step until no new node is activated, the propagation ends, and all nodes in the active state are returned.

4. Algorithm

4.1. Quantum-behaved particle swarm optimization

PSO, one of the most widely used swarm intelligence algorithms, was originally proposed by Kennedy and Eberhart.[38] PSO is often used to solve optimization problems due to its effectiveness and robustness. The original updating rules for PSO can be described as follows:

$$V_i^{t+1} = wV_i^t + c_1r_1(\mathrm{Pbest}_i - X_i^t) + c_2r_2(\mathrm{Gbest} - X_i^t), \tag{4}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}, \tag{5}$$

where w is the inertia weight, c_1 is the individual learning factor, c_2 is the social learning factor, and r_1 and r_2 are random numbers drawn uniformly from $[0,1]$. $X_i=(x_1,x_2,\ldots,x_d)$ and $V_i=(v_1,v_2,\ldots,v_d)$ represent the position vector of the ith particle in the dth dimension and its corresponding velocity vector, respectively. Pbest_i denotes the historical best position of particle i and Gbest is defined as the best position in the whole population. More specifically, the first part wV_i^t in Eq. (4) reflects the effect of the particle's velocity at time t on its new velocity at time $t+1$, and the second part $c_1r_1(\mathrm{Pbest}_i - X_i^t)$ and the third part $c_2r_2(\mathrm{Gbest} - X_i^t)$ imply that the particle learns information from its corresponding historical optimal position and global optimal position of the population, respectively.

In recent years, many strategies have been proposed to enhance the performance of PSO, such as Lévy flight,[39] multi-swarm cooperative approach,[40] and fitness landscape features.[41] Inspired by quantum mechanics theory and trajectory analysis,[42] the ideology of quantum parallel mechanics is introduced into the framework of PSO to improve the performance of the algorithm, termed the quantum particle swarm optimisation algorithm, which outperforms traditional PSO in terms of exploration capability but with fewer control parameters. In QPSO, the position of the ith particle is updated according to the following equations:

$$P_i^t = \varphi \text{Pbest}_i^t + (1-\varphi)\text{Gbest}, \tag{6}$$

$$X_i^{t+1} = P_i^t \pm \beta |\text{mbest}^t - X_i^t| \ln\left(\frac{1}{u}\right), \tag{7}$$

where Pbest_i and Gbest denote the historical best position and the global best position of the ith particle, respectively. P_i is defined as a local attractor. According to Eq. (6), it can be seen that the local attractor P_i is located in a hyper-rectangle with Pbest_i and Gbest as vertices. φ and u are random numbers in the interval $(0,1)$, β is a contraction-expansion factor, and mbest is the average of the best positions of all particles. If $u < 0.5$, then the minus symbol "$-$" is selected in Eq. (7), otherwise the plus sign "$+$" will be selected. The framework of QPSO is given in Algorithm 1.

Algorithm 1 The framework of QPSO.

Initialize each particle's position, Pbest, Gbest
while $T < T_{\max}$ **do**
 for each particle **do**
  Calculate local attractor P_i using Eq. (6)
  Compute particle's position using Eq. (7)
  Update Pbest and Gbest
 end for
end while
Return the best position Gbest.

4.2. Lévy flight

In nature, most animals generally combine frequent short-distance wandering with occasional long-distance travel when foraging food. As a random walk process, Lévy flight mainly consists of frequent local exploitation and occasional global exploration of the search space. The application of this mechanism to swarm-based intelligence algorithms can enhance the global search capability of the metaheuristics and prevent them from falling into premature convergence. By combining the characteristics of Lévy flight, some researchers have introduced hybrid swarm intelligence optimization algorithms[36] with promising results. The following are the rules to calculate the inherent step lengths of Lévy flight:

$$l = \frac{\mu}{|\nu|^{1/m}}, \tag{8}$$

where l represents the random step. For the parameter setting, $1 < m \leq 3$, and $\mu \sim N(0, \sigma_\mu^2)$, $\upsilon \sim N(0, \sigma_\upsilon^2)$. υ and υ are random numbers obeying Gaussian distribution, while σ_μ and σ_υ satisfy the following equations:

$$\sigma_\mu = \left\{\frac{\Gamma(1+m)\sin(0.5m\pi)}{\Gamma[0.5+(1+m)]2^{0.5(m-1)}}\right\}^{\frac{1}{m}}, \quad \sigma_\upsilon = 1, \tag{9}$$

where $\Gamma(\cdot)$ denotes Gamma function. Figure 1 shows the trajectory of an individual wandering randomly in the search space under the Lévy flight mechanism.

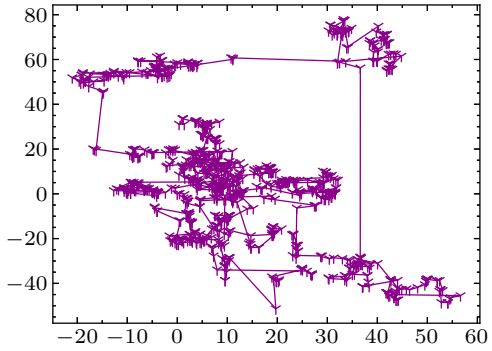


Fig. 1. The trajectory of an individual after flying 1000 steps under the Lévy flight in two dimensions.

4.3. Proposed algorithm

The whole framework of the proposed algorithm for IM is outlined in Algorithm 2. The algorithm is divided into three main stages: (i) initialize the particles in the population; (ii) update the positions of the particles in the population according to the rules of DQPSO; and (iii) apply the Lévy flight strategy on the Gbest and finally output the optimal solution.

Algorithm 2 The framework of the proposed algorithm.

Input Graph $G = (V, E)$, size of particle swarm n, size of seed set k, number of iterations $T_{\max}$ and the contraction-expansion coefficient β_1 and β_2.
Initialize particle position vectors X and Pbest
Initialize historical diversity value HDV $\leftarrow 0$
Initialize stages of algorithm evolution Stage $\leftarrow 1$
Initialize iterator $T \leftarrow 0$
Compute the shortest path length matrix M for graph G
Compute fitness(X) and fitness(Pbest)
Select out the initial global best position vector Gbest
while $T < T_{\max}$ **do**
 Update the stage of the evolution Stage and identify the algorithm to be selected Algorithm
 switch Algorithm **do**
  case Algorithm == "DQPSO" **do**
   Apply the DQPSO algorithm to update X
   Update Pbest and Gbest
  end case
  case Algorithm == "DLF" **do**
   Apply the DLF algorithm to update Gbest
  end case
 end switch
 $T \leftarrow T + 1$
end while
Output Output the best position Gbest as the seed set S.

4.3.1. Initialization

During the initialization phase, a degree-based initialization strategy similar to that of DPSO is employed. First, all particles in the population select the top k nodes with the greatest degree in the graph G. At the same time, to guarantee the diversity of the population, a random replacement operation on the nodes in each particle is performed, i.e., when the random value is greater than 0.5, the corresponding node will be replaced by any node in the graph G that has not been selected as a candidate node in that particle. The same approach is utilized when initializing Pbest. The detailed initialization procedure is given in Algorithm 3.

Algorithm 3 Initialization.

Input Graph $G=(V,E)$, the size of particle swarm n, size of seed set k.
for each $i \le n$ **do**
  $X_i \leftarrow \text{degree}(G,k)$
  for each element $x_{ij} \in X_i$ **do**
    if random>0.5 **then**
      $x_{ij} \leftarrow \text{replace}(x_{ij},N)$
    end if
  end for
end for
Output The initial position vector X.

4.3.2. Discrete quantum-behaved particle swarm optimization algorithm

The original QPSO algorithm is only applicable to solve optimization problems with continuous space. Therefore, the update rules are redesigned in discrete form to solve the IM problem. The new evolutionary mechanism of DQPSO is defined as

$$X_i^{t+1} = P_i^t \oplus (\text{mbest}^t \cap X_i^t)\beta \ln(1/u), \tag{10}$$

where u is a random value between $1/\text{e}$ and 1, X_i^t is the position vector of the ith particle at iteration t, β represents contraction-expansion coefficient, and $\beta\ln(1/u)$ is redefined as the crossover probability, as described by the following equation:

$$\beta = (\beta_1 - \beta_2)\left(\frac{T_{\max}-t}{T_{\max}}\right)^2 + \beta_2 \tag{11}$$

for $\beta_1 = 1$, $\beta_2 = 0.5$. $T_{\max}$ represents the maximum number of iterations. From Eqs. (10) and (11), it is obvious that β generally varies nonlinearly from 1 to 0. When β reaches a large value in the first few steps, the crossover ability of the particles becomes stronger, which indicates that the particles have stronger exploration ability. In the later stages, the crossover ability of the particles gradually becomes weaker and ostensibly the particles gradually tend to converge. However, since $\ln(1/u)$ is a random value, there is also a lower probability of obtaining a stronger crossover ability, which to some extent increases the diversity of the population.

In Eq. (10), P_i^t is given by

$$P_i^t = \lceil k\varphi \rceil \text{Pbest}_i^t + (k - \lceil k\varphi \rceil)\text{Gbest}, \tag{12}$$

where φ is a random value drawn from $[0,1]$, $k\varphi$ is rounded upward to determine the number of nodes to be randomly selected in Gbest or Pbest, and then the selected nodes are merged to obtain the local attractor P_i. When selecting nodes at random, it is important to ensure that the selected nodes cannot be duplicated. The specific operation is shown in Fig. 2, assuming that $\text{Pbest}_i = \{1,5,10,17,20\}$, $\text{Gbest} = \{3,5,11,14,19\}$, $k=5$, $\varphi = 0.36$, then $k\varphi = 2$; therefore, two nodes in Pbset and three nodes in Gbest need to be taken for combination.

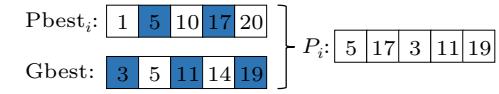


Fig. 2. Illustration of how to obtain local attractor P_i. The blue boxes represent the nodes to be randomly selected.

In the original QPSO algorithm, mbest is the average of the best positions of all of the particles. When mbest takes this case, it does not guide the particles very well, and therefore mbest is redefined as the average of the optimal positions of the top three particles. First, the top three particles are divided into two portions, the first portion includes the common nodes belonging to the three particles, and the second portion maintains the remaining nodes in the particles. All of the nodes in the first portion are then selected into Mbest. Finally, the nodes are randomly selected from the second component until there are k nodes in the Mbest. A detailed illustration of this operator is shown in Fig. 3. It is assumed that the top three historical optimal positions are $\text{Pbest}_1 = \{1,3,4,7,11\}$, $\text{Pbest}_2 = \{1,4,6,8,10\}$, $\text{Pbest}_3 = \{1,5,4,8,11\}$, the identical nodes are then $\{1,4\}$, and the remaining nodes are $\{3,10,11,6,8,7,5\}$, finally $\text{Mbest} = \{1,4,8,5,10\}$ is obtained by the restructuring mechanism.

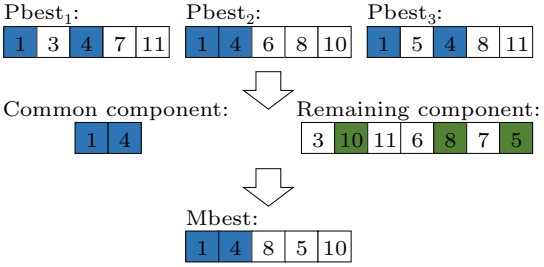


Fig. 3. Illustration of the calculation of Mbest. The blue boxes represent common nodes belonging to all three particles and the green boxes represent randomly selected nodes.

The operator $\cap$ in Eq. (10) is a logical operator that is defined as a similar intersection operation to determine whether there are different elements in Mbest and X_i. When a node in X_i is a unique node to Mbest, the crossover probability $\beta\ln(1/u)$ is calculated. If $\beta\ln(1/u)$ is greater than a random value drawn from $[0,1]$, then a node in Mbest that is not in X_i

is selected for the intersection exchange operation; otherwise, no operation is performed on that node. The $\oplus$ operation is the process of greedy search of the local attractor P subject to the new particle position vector X_i obtained by the crossover operation. In this process, the nodes in the local attractor P_i that are identical to those in X_i are removed and then treated as a pool of candidate nodes. For each node in X_i, an arbitrary node from the candidate node pool is selected to replace it. Meanwhile, the selected node is removed from the candidate node pool. When the position of the node after being replaced is better than the previous position, then any node from the candidate node pool continues to be selected for replacement; otherwise, the same operation is performed for the next node in X_i. It should be noted that the operation is terminated when there is no node in the candidate pool. The framework of the DQPSO algorithm for IM is presented in Algorithm 4.

Algorithm 4 DQPSO algorithm framework.

Input Particle position vectors X, the size of particle swarm n, size of seed set k, particle local attractor position vectors P_attr, iterator T, the mean of the best position of the top three particles Mbest, the number of iterations $T_{\max}$, and the contraction-expansion coefficient β_1 and β_2.
for each $i \leq n$ **do**
 if $X_{ij} \notin M$_best **do**
  Compute the probability of mutation p_{mu}
  if $p_{\mathrm{mu}} >$ rand **do**
   $X_i \leftarrow \mathrm{replace}(X_{ij}, M_\mathrm{best})$
  end if
 end if
end for
for each $i \leq n$ **do**
 candidate_pool $= \emptyset$
 if $P_\mathrm{attr}_{ij} \notin X_i$ **do**
  cand_pool $\leftarrow$ cand_pool $\cup P_\mathrm{attr}_{ij}$
 end if
 $X_i' \leftarrow X_i$
 for each $j \leq k$ **do**
  if cand_pool $== \emptyset$ **do**
   break
  end if
  Flag $\leftarrow$ False
  while Flag $==$ False **do**
   if cand_pool $== \emptyset$ **do**
    break
   end if
   $X_{ij}' \leftarrow \mathrm{replace}(X_{ij}', \mathrm{cand_pool})$
   if $\mathrm{fitness}(X_i') > \mathrm{fitness}(X_i)$ **do**
    $X_{ij} \leftarrow X_{ij}'$
   else do
    Flag $\leftarrow$ True
   end if
  end while
  $X_{ij}' \leftarrow X_{ij}$
 end for
 $X_i \leftarrow X_i'$
end for
Output Output particle position vectors X.

4.3.3. Algorithm transformation

To avoid the DQPSO algorithm converging swiftly to a suboptimal global solution, a metric to assess the diversity of the population is conceived. The diversity metric can be formulated as follows:

$$\mathrm{Diversity} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\left(\frac{X_i \bigcap \mathrm{Mbest}}{k}\right)^2 + \left(\frac{X_i \bigcap \mathrm{Pbest}_i}{k}\right)^2}, \quad (13)$$

where n is the size of the particle swarm, k is the size of the seed set, and the $\cap$ operation returns the number of identical nodes in the two position vectors. It can be seen from Eq. (13) that the diversity value becomes larger and the diversity of the population decreases with the evolution of the algorithm. When the diversity value of the population is less than its historical, it indicates that the population has converged to a stagnation. In this case, the algorithm will perform the second stage, i.e., discrete Lévy flight, to identify a more optimal position. The framework of updating stage and identifying algorithm is given in Algorithm 5.

Algorithm 5 Update Stage and identify Algorithm.

Input. Particle position vectors X, particle best position vectors Pbest, evolution stage Stage, and historical diversity value HDV.
switch Stage **do**
 case Stage $==$ 1 **do**
  Compute Mbest
  Compute P_attr
  Compute the current diversity value CDV
  if CDV $-$ HDV > 0 **do**
   HDV $=$ CDV
   Algorithm $=$ "DQPSO"
  end if
  else do
   Stage $= 2$
   Algorithm $=$ "DLF"
 end case
 case Stage $==$ 2 **do**
  Algorithm $=$ "DLF"
 end case
end switch
Output Output the stage of the evolution Stage and identify the algorithm to be selected Algorithm.

4.3.4. Discrete Lévy flight

The original Lévy flight is not available to solve discrete IM problem directly. Therefore, a discrete Lévy flight mechanism based on the shortest path length of the network is proposed.

Since the previous stage evolves through the DQPSO algorithm, the population tends to remain in an optimal state. Therefore, there is no need to perform discrete Lévy flight for all of the particles in the population but only for the Gbest. The shortest path length matrix M of graph G is first obtained in the initialization phase, and then Lévy flight step length l is

derived from Eq. (7) and rounded upward. During the searching process, each node in Gbest will be replaced with a randomly selected node from the candidate pool, which is composed of nodes in Gbest with the same shortest path length in the network and the same flight step length l but not the same as in Gbest. When the step flight length l is greater than the maximum shortest path length of that node in the network, the flight length l takes the maximum shortest path length of that node in the network. The detailed replacement process is the same as the $\oplus$ operation mentioned earlier. The framework of the DLF algorithm for IM is described in Algorithm 6.

Algorithm 6 DLF framework of algorithm.

Input The best position Gbest and the shortest path length matrix M.
Gbest$'$ $\leftarrow$ Gbest
for each $i \leq k$ **do**
  compute the step size of the Lévy flight l
  cand_pool $\leftarrow M(\text{Gbest}(i), l)$
  if cand_pool $==$ $\emptyset$ **do**
    break
  end if
  Flag $\leftarrow$ False
  while Flag $==$ False **do**
    if cand_pool $==$ $\emptyset$ **do**
      break
    end if
    Gbest$'_i \leftarrow$ replace(Gbest$'_i$, cand_pool)
    if fitness(Gbest$'$) $>$ fitness(Gbest) **do**
      Gbest $\leftarrow$ Gbest$'$
    else do
      Flag $\leftarrow$ True
    end if
  end while
  Gbest$'$ $\leftarrow$ Gbest
end for
Output Output the best position Gbest$'$.

5. Experiment and analysis

5.1. Network data sets and baselines

To verify the performance of the proposed DTMO on the IM problem, we conducted extensive experiments on six real-world networks that were collected from NR with the topological characteristics shown in Table 1, where $\langle k \rangle$ represents the average node degree, C is the average clustering coefficient, and AC represents the assortativity coefficient. Figure 4 shows the node degree distribution of the networks. The results of influence spread are compared with several state-of-the-art methods, including the degree centrality (DC) based on the network topology, discrete moth-flame optimization (DMFO),[28] DDSE,[26] DPSO,[24] learning automata-based discrete particle swarm optimization (LAPSO),[27] cost-effective lazy forward (CELF)[14] algorithm based on greedy strategy, and layered gravity bridge algorithm (LGB).[22]

Table 1. Topological characteristics of the six social networks.

Networks	$\lvert V\rvert$	$\lvert E\rvert$	$\langle k \rangle$	C	AC
Delaunay	1024	3056	5.9688	0.4356	−0.1153
Blog	3982	6803	3.4169	0.2838	−0.133
Ca-GrQc	4158	13422	6.456	0.5569	0.6392
PGP	10680	24316	4.5536	0.2659	0.2382
Sex	15810	38540	4.8754	0	−0.1145
AstroPh	18772	198110	21.107	0.6306	0.2051

5.2. Parameter configuration

To test the scalability of the algorithm, the size of the seed set k was set to 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50, respectively, the maximum iterations $T_{\max}$ were 100 and the population size was chosen as 100. In DMFO, the amplification factor was set to $c = 4$, the mutation probability was 0.1, and the coefficient ω was 0.8. In DPSO and LAPSO, the learning factors were set to $c_1 = c_2 = 2$, the inertia weight was 0.8, and additionally the values of the reward and punishment parameters were set to $a_r = b_p = 0.6$ in LAPSO. In DDSE, the probabilities of mutation, crossover, and diversity operations were set to 0.1, 0.4, and 0.6, respectively. The number of Monte Carlo simulations for CELF was set to 10000. In addition, 1000 Monte Carlo simulations were performed on the optimal set of seeds obtained by all of the other algorithms to obtain the average spreading coverage under the IC model with propagation probability $p = 0.01$.

5.3. Comparison of the LIE evaluation

To validate that DQPSO and DLF play a desirable effect on their respective phases, we compared the fitness values of the obtained optimal seed sets of DTMO, DQPSO, and DPSO. Figure 5 shows the average LIE of different seed size k on the six networks.

In the Delaunay network, as shown in Fig. 5(a), all three algorithms perform competitive results over the entire interval, and a detail view at $k = 50$ shows that DTMO gains a better marginal gain. In Blog and CaGrQc, the algorithms gradually show visible gaps as the seed number k increases and the gap gradually becomes more evident as the network's scale increases. In Figs. 5(d) and 5(e), it can be seen that DPSO is highly prone to suboptimal solutions, while DTMO and DQPSO have no evident instability, which indicates that the proposed algorithm has strong robustness. It is worth mentioning that, except for the Ca-GrQc network, the fitness values of DTMO and DQPSO are not significantly different throughout the interval of k. In contrast, in Fig. 5(c), there is a significant gap between the algorithm DTMO and DQPSO at $k = 50$. This indicates that the random wandering strategy in the second stage of the algorithm compensates for the deficiency of the DQPSO algorithm falling into premature convergence. Overall, both the two-stage DTMO and the separate DQPSO achieved satisfactory results in all six networks.

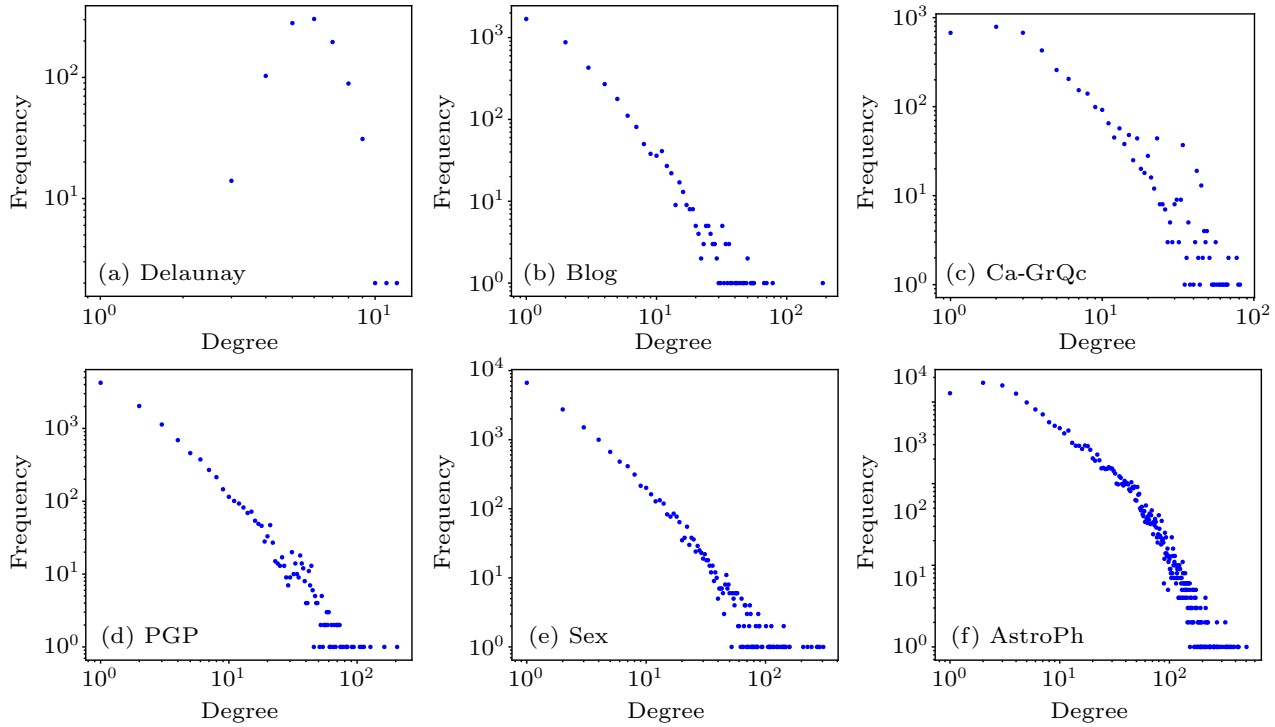


Fig. 4. Node degree distribution of the six networks.

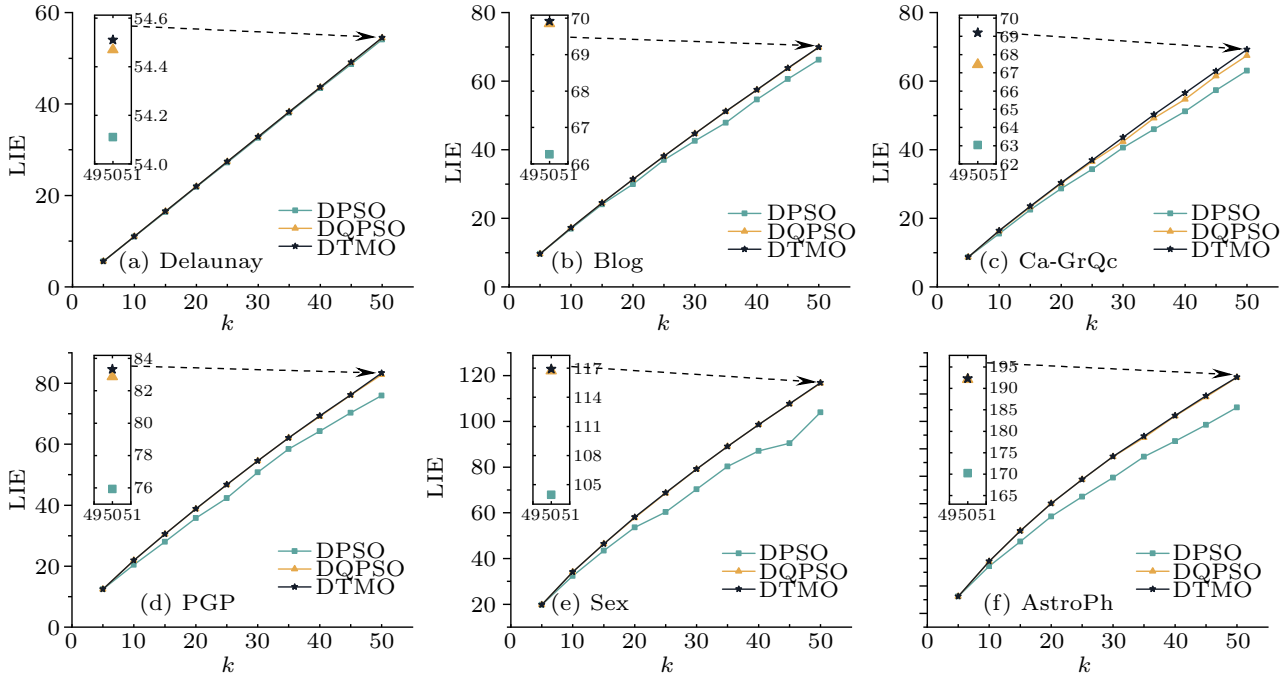


Fig. 5. Comparison of LIE optimization of the three algorithms in the six social networks at different seed size k under $p = 0.01$.

5.4. Performance comparison of the typical algorithms

To further validate the ability of the proposed DTMO algorithm to solve the IM problem, six state-of-the-art methods were selected to compare with DTMO in terms of the number of nodes activated in the network under the IC model. As shown in Fig. 6, the spread size achieved by DTMO shows that it can return satisfactory propagation in the six networks. Moreover, it can be seen that DTMO has the most stable performance in different scenarios.

More specially, in the Delaunay network, it can be observed that the algorithms achieve similar influence spread throughout the range of $5 \leq k \leq 50$, and the proposed DTMO algorithm achieves the best result. In the Blog network shown in Fig. 6(b), comparable results are obtained for all six algorithms except DMFO on the interval of $15 \leq k \leq 25$. There is a significant decline in the effectiveness of DPSO when $k \geq 30$. At $k = 50$, DDSE outperforms the others by a significant margin, while DTMO comes in a close second.

Figure 6(c) shows that CELF, DMFO, LGB, and DTMO can maintain good momentum when $k \geq 15$. LAPSO and LGB algorithms achieve similar performance over the entire interval, while the other algorithms show different levels of performance degradation. At $k = 45$ and $k = 50$, both DTMO and DMFO outperformed the CELF and achieved the best influence spread. In addition, DTMO and DMFO achieved one optimal result each, indicating that both DTMO and DMFO have strong competitiveness in this network. Figure 6(d) shows that the ranking of all algorithms remains essentially constant over the entire range of seed size, with CELF and DTMO achieving comparable performance, and LAPSO coming in second. It should be noted that in Figs. 6(c) and 6(d), both DDSE and DC showed lower efficiency. This is probably due to the fact that in both networks only the one-hop neighbors of the candidate nodes are evaluated, which leads to the problem of overlapped influence propagation in the propagation process.

In Fig. 6(e), similar experimental results were obtained by LAPSO, CELF, DC, LGB, and DTMO. More specifically, LAPSO performed the best in the interval of $15 \leq k \leq 45$, while CELF and DTMO were entangled with each other in the influence size. At $k = 50$, DTMO, LGB, and LAPSO obtained the same results. It is worth mentioning that the topology of the network is a star structure, so DC can achieve satisfactory results. Meanwhile, DC degenerated at $k = 50$ because it suffers from the influence overlap problem as the number of selected nodes increases. In the AstroPh network of Fig. 6(f), DMFO and DTMO have achieved relatively high performance but were slightly inferior to CELF.

In summary, DTMO achieved comparable or even better results than CELF in most cases, indicating that DTMO can effectively solve the IM problem. DC and DDSE performed poorly in some networks because they only evaluated the one-hop neighbors of the nodes, resulting in inaccurate evaluation of influence. Although DPSO was more accurate in evaluating candidate nodes, the algorithm tended to be stagnant because its local search strategy easily returns the local optimum. LGB offers a dramatic improvement in influence dissemination when compared to traditional centrality-based approaches. LAPSO and DMFO achieved desirable results in some scenarios. This indicates that the improvement of PSO and the searching strategy in DMFO are very effective, and the respective evaluation functions can accurately evaluate the influence of the nodes.

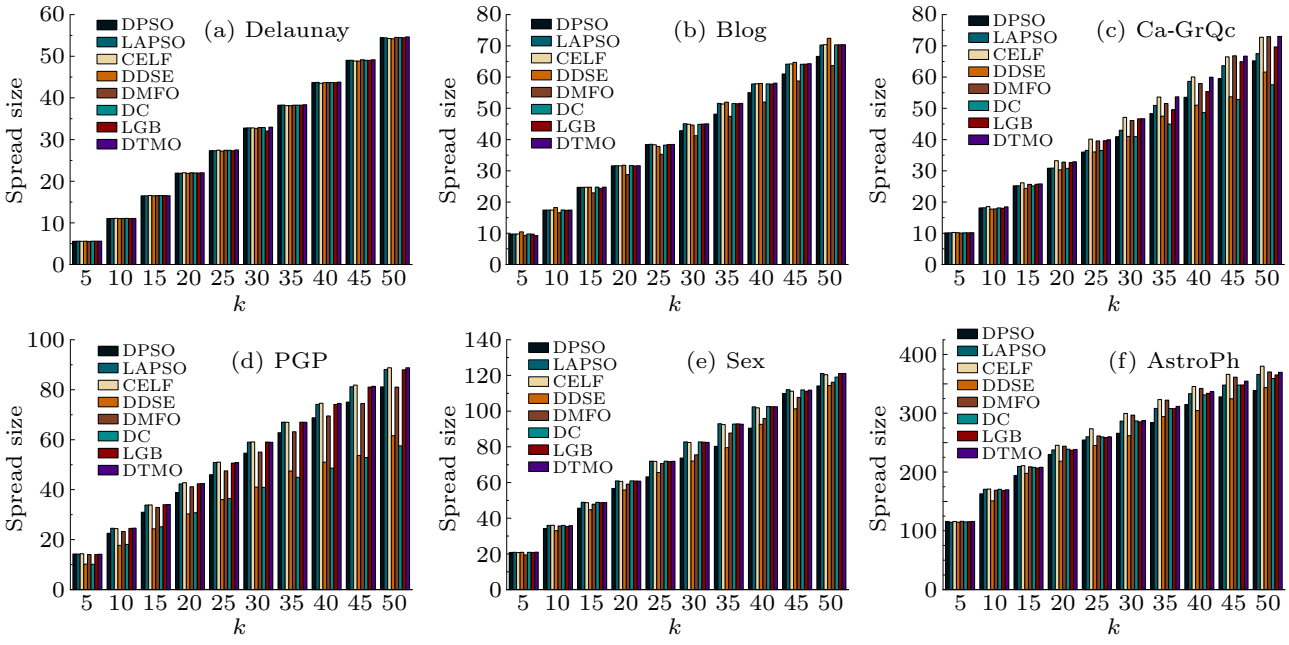


Fig. 6. Comparison of the influence spread size of DTMO against six other algorithms under the IC model ($p = 0.01$) on the six networks.

5.5. Statistical tests

To demonstrate the effectiveness of the proposed DTMO algorithm in identifying influence spreaders, a statistical analysis was employed to test the statistical significance of the performance differences of the six algorithms on the six networks, the results are reported in Table 2. In all six networks, the scenarios of $k = 10, 20, 30, 40$, and 50 were chosen as independent problems, and hypotheses were tested for each scenario. Wilcoxon rank sum test at the confidence level of 0.05 was conducted to show the superior performance of DTMO over the other six algorithms.

From the statistical results, it can be seen that the influence propagation obtained by the DTMO algorithm is better than DPSO algorithm on the whole selected interval. DDSE, DMFO, DC, and LGB have similar performance, while LAPSO has slightly better performance. At the same time, it can give a similar performance guarantee when compared with CELF.

Table 2. Statistical results of the Wilcoxon test for the seven algorithms at $\alpha = 0.05$.

DTMO vs.	k	$N+$	$N-$	Z	p-value
DPSO	10	6	0	−2.201	0.028
	20	6	0	−2.201	0.028
	30	6	0	−2.201	0.028
	40	6	0	−2.201	0.028
	50	6	0	−2.201	0.028
LAPSO	10	3	3	−0.105	0.917
	20	4	2	−1.363	0.173
	30	3	3	−0.734	0.463
	40	6	0	−2.201	0.028
	50	5	1	−1.992	0.046
CELF	10	2	4	−0.734	0.463
	20	2	4	−1.363	0.173
	30	3	3	−0.734	0.463
	40	3	3	−0.314	0.753
	50	5	1	−0.943	0.345
DDSE	10	5	1	−1.572	0.116
	20	5	1	−1.728	0.075
	30	6	0	−2.201	0.028
	40	6	0	−2.201	0.028
	50	5	1	−1.782	0.075
DMFO	10	6	0	−2.201	0.028
	20	5	1	−0.943	0.345
	30	5	1	−0.943	0.345
	40	5	1	−1.363	0.173
	50	5	1	−1.572	0.116
DC	10	2	4	−0.105	0.917
	20	3	3	−0.314	0.753
	30	5	1	−1.572	0.116
	40	5	1	−1.992	0.046
	50	6	0	−2.201	0.028
LGB	10	6	0	−2.201	0.028
	20	5	1	−1.572	0.116
	30	4	2	−0.734	0.463
	40	6	0	−2.201	0.028
	50	5	1	−1.992	0.046

6. Conclusion

In this paper, a DTMO algorithm combining QPSO with Lévy is proposed to solve the IM problem. In the first stage of the algorithm, the evolutionary rules of the redefined DQPSO are used to update the particle positions to make the particles fly to a more optimal position. At the same time, population diversity is defined to determine whether the particles in the population converge to the optimal. When the particles in the population are in convergence, the second stage of the algorithm is performed to obtain the global optimal solution by updating the optimal positions of the particles in the population using the redesigned DLF mechanism. Finally, experiments with other well-known algorithms on six real networks show that the proposed algorithm obtains comparable or even slightly superior results to CELF but with less computational cost and obtains significantly better results when compared to the other algorithms.

Acknowledgments

Project supported by the Zhejiang Provincial Natural Science Foundation (Grant No. LQ20F020011), the Gansu Provincial Foundation for Distinguished Young Scholars (Grant No. 23JRRA766), the National Natural Science Foundation of China (Grant No. 62162040), and the National Key Research and Development Program of China (Grant No. 2020YFB1713600).

References

[1] Zamani E D and Spanaki K 2023 *J. Bus. Res.* **154** 113311
[2] Nabity-Grover T, Cheung C M K and Bennett Thatcher J 2023 *J. Bus. Res.* **154** 113310
[3] Hu X J, Qiu J, Zhao J and Li Y 2023 *J. Retail. Consum. Serv.* **70** 103142
[4] Xiao Y P, Huang Z, Li Q , Lu X Y and Li T 2022 *IEEE Trans. Knowl. Data Eng.* **35** 4682
[5] Domingos P and Richardson M 2001 *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (San Francisco: California) p. 57
[6] Kempe D, Kleinberg J and Tardos E 2003 *Proceedings of the ninth ACM SIGKDD international confer-ence on Knowledge discovery and data mining* (Washington: D.C.) p. 137
[7] Freeman L C 1978 *Soc. Netw.* **1** 215
[8] Brin S and Page L 1998 *Comput. Netw. ISDN Syst.* **30** 107
[9] Alshahrani M, Zhu F X, Sameh A, Mekouar S and Huang S 2020 *Inform. Sci.* **527** 88
[10] Bozorgi A, Haghighi H, Sadegh Zahedi M and Rezvani M 2016 *Inform. Proc. Manag.* **52** 1188
[11] Cai F, Qiu L R Kuai X K and Zhao H S 2019 *IEEE Access* 7 152115
[12] Zareie A, Sheikhahmadi A and Jalili M 2020 *Expert Syst. Appl.* **142** 112971
[13] Li H, Zhang R S, Zhao Z L, Liu X and Yuan Y N 2021 *Appl. Intell.* **212** 118702
[14] Leskovec J, Krause A and Guestrin C, Faloutsos C, VanBriesen J and Glance N 2007 13*th International Conference on Knowledge Discovery and Data Mining* (San Jose: CA) p. 420
[15] Goyal A, Lu W and Lakshmanan L V S 2011 *Proceedings of the* 20*th international conference companion on World wide web* (Hyderabad: India) p. 47
[16] Zhang S Q, Zeng X X and Tang B 2021 *Inf. Syst.* **102** 101828
[17] Chen W, Wang Y J and Yang S Y 2009 *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining* (Paris: France) p. 199
[18] Zareie A and Sheikhahmadi A 2018 *Expert Syst. Appl.* **93** 200
[19] Wang Y F, Dong W Y and Dong X S 2018 *Fut. Gener. Comput. Syst.* **88** 755
[20] Li W M , Zhong K X , Wang J J and Chen D H 2021 *Expert Syst. Appl.* **169** 114207
[21] Meng L, Xu G Q and Yang P L and Tu D Q 2022 *J. Comput. Sci.* **60** 101591
[22] Li Q, Cheng L, Wang W,Li X H,Li S D and Zhu P C 2023 *Appl. Math. Comput.* **442** 127721
[23] Jiang Q Y, Song G J, Gao C,Wang Y, Si W J and Xie K Q 2011 *Proceedings of the Twenty-Fifth AAAI Conference on Articial Intelligence* (San Francisco, California) p. 7
[24] Gong M G, Yan J N and Shen B,Ma L J and Cai Q 2016 *Inform. Sci.* **367** 600
[25] Tang J X, Zhang R S, Yao Y B, Zhao Z L, Wang P, Li H and Yuan J L 2018 *Knowl. Based Syst.* **160** 88
[26] Cui L Z, Hu H X, Yu S, Yan Q, Ming Z, Wen Z K and Lu N 2018 *J. Netw. Comput. Appl.* **103** 119
[27] Singh S S, Kumar A, Singh K and Biswas B 2019 *Appl. Soft Comput.* **82** 105554
[28] Wang L, Ma L and Wang C and Xie N G 2021 *IEEE Trans. Evol. Comput.* **25** 1091
[29] Weskida M and Michalski R 2019 *J. Comput. Sci.* 31 77
[30] Sun J, Feng B and Xu W B 2004 *IEEE Conf. Cyber Inte. Syst.* **1** 111
[31] Yang C X, Zhang J and Tong M S 2021 *IEEE Trans. Antennas Propag.* **69** 9
[32] Bajaj A and Abraham A 2022 2022 *IEEE Cong. Evol. Comput.* (CEC) 1

[33] Dian S Y, Zhong J N, Guo B, Liu J X and Guo R 2022 *Expert Syst. Appl.* **208** 118256
[34] Lu X L and He G 2021 *Appl. Soft Comput.* **99** 106894
[35] Li L L, Jiao L C and Zhao J Q Shang R H and Gong M G 2017 *Patt. Reco.* **63** 1
[36] Zhou B H and Lei Y R 2021 *Appl. Soft Comput.* **111** 107717
[37] Pei S, Muchnik L, Andrade J S, Zheng Z M and Makse H A 2014 *Sci. Rep.* **4** 5547
[38] Kennedy J and Eberhart R 1995 *Proceedings of ICNN'95 - International Conference on Neural Networks* **4** 1942
[39] Jensi R and Jiji G W 2016 *Appl. Soft Comput.* **43** 248
[40] Lu J W, Zhang J and Sheng J N 2022 *Swarm Evol. Comput.* **69** 100989
[41] Dennis C, Ombuki-Berman B M and Engelbrecht A 2021 2021 *IEEE Cong. Evol. Comput.* (CEC) **10** 2289
[42] Clerc M 1999 *Proceedings of the* 1999 *Congress on Evolutionary Computation-CEC*99 **3** 1951