Expert Systems With Applications 142 (2020) 112971

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Identification of influential users in social network using gray wolf optimization algorithm

Ahmad Zareie[c], Amir Sheikhahmadi[a,*], Mahdi Jalili[b]

[a] Department of Computer Engineering, Islamic Azad University Sanandaj Branch, Sanandaj, Iran
[b] School of Engineering, RMIT University, Melbourne, Australia
[c] School of Computer Science, University of Manchester, UK

ARTICLE INFO

Article history:
Received 17 May 2019
Revised 7 September 2019
Accepted 19 September 2019
Available online 21 September 2019

Keywords:
Gray wolf optimizer
Influence maximization
Social networks
Spreading process
Viral marketing

ABSTRACT

A challenging issue in viral marketing is to effectively identify a set of influential users. By sending the advertising messages to this set, one can reach out the largest area of the network. In this paper, we formulate the influence maximization problem as an optimization problem with cost functions as the influentiality of the nodes and the distance between them. Maximizing the distance between the seed nodes guarantees reaching to different parts of the network. We use gray wolf optimization algorithm to solve the problem. Our experimental results on three real-world networks show that proposed method outperforms state-of-the-art influence maximization algorithms. Furthermore, it has lower computational time than other meta-heuristic methods.

1. Introduction

Survival of many business firms depends on the presentation and sale of their products. For this purpose, they are always looking for ways to effectively represent their products in the marketplace. Word of mouth and propagating the advertisement messages in an appropriate context by people selected properly, may achieve this goal. Online social networks are appropriate choices for propagating advertisements of products. Engrossing environments of these networks have attracted many users, which are on increase every day (Kimura, Saito, Nakano & Motoda, 2010; Sheikhahmadi, Nematbakhsh & Zareie, 2017; Zareie, Sheikhahmadi & Jalili, 2019a). Viral marketing can much benefit from properties of online social networks, where a set of users, referred to as seed set, is selected to initiate spreading the messages on the network. Due to the limitation of advertising budget of firms, there is only a limited number of members of the seed (Chevalier & Mayzlin, 2006; Zareie, Sheikhahmadi & Khamforoosh, 2018). A question arises here that is how to effectively identify the most influential users to place them into the seed set. This problem is known as Influence Maximization (IM) problem, that is defined as to identify a set of k users on which the spreading process is initiated (Lu et al., 2017; Sheikhahmadi, Nematbakhsh & Shokrollahi, 2015). Although in some approaches, e.g. (Sheikhahmadi et al., 2017; Zareie, Sheikhahmadi & Jalili, 2019b), users' behavior and interests have been taken into account as important factors in the spreading process, such information are not always available in real networks. Thus, we often have only network structural information available to identify influential users.

A number of diffusion models (Huang, Lee, Wen & Sun, 2013; Jalili & Perc, 2017; Nowzari, Preciado & Pappas, 2016) have been proposed to simulate the spreading process and measure the influentiality of the seed set. Due to large search space of the problem, assessing all possible subsets to find the optimal seed set is an NP-hard problem (Kempe, Kleinberg & Tardos, 2003).

Real networks are often large-scale, and thus applying diffusion models to measure the influentiality of all users can be a time consuming process. A possible way to identify the most influential users is to choose the most central nodes. Structural centrality can be obtained using various measures, such as degree (Freeman, 1978), betweenness (Freeman, 1977), closeness (Sabidussi, 1966), K-shell (Kitsak et al., 2010), and hierarchical K-shell (Zareie & Sheikhahmadi, 2018). However, structural centrality measures often do not result in near optimal solution to the problem (Bao, Liu & Zhang, 2017; Zareie et al., 2018). Some other research studies have formulated IM problem as an optimization problem, used greedy, heuristic, and meta-heuristic methods to

* Corresponding author.
E-mail addresses: ahmad.zareie@postgrad.manchester.ac.uk (A. Zareie), sheikhahmadi@eng.ui.ac.ir (A. Sheikhahmadi), Mahdi.jalili@rmit.edu.au (M. Jalili).

https://doi.org/10.1016/j.eswa.2019.112971
0957-4174/

solve it. Greedy methods (Kempe et al., 2003; Wang, Cong, Song & Xie, 2010) select a solution close to the optimal solution by using a diffusion model. Although they find highly accurate solutions, they suffer from high computational complexity, making them impractical on large networks (Gong, Yan, Shen, Ma & Cai, 2016). Heuristic methods (Bao et al., 2017; Zareie et al., 2018) proposed for IM problem compromise the accuracy by reducing computational complexity. They are also likely to trap in a local optima (Cui et al., 2018). A number of recent works (Cui et al., 2018; Gong et al., 2016) proposed evolutionary optimization methods to avoid this. These research studies have been carried out to identify near-optimal solution of the problem.

To find a near-optimal set of the influential users, one needs to first measure the influence of users in an effective manner, and then find the most influential users. In this work we first use a metric based on entropy to measure influentiality of users. We then formulate the problem as an optimization problem and use gray wolf optimization method (Mirjalili, Mirjalili & Lewis, 2014) to solve it. In comparison to other evolutionary optimization algorithm, this algorithm has less tunable parameters, and can search the problem space with less computational complexity and faster convergence time (Jayakumar, Subramanian, Ganesan & Elanchezhian, 2016; Pradhan, Roy & Pal, 2016).

The rest of the paper is organized as follow. Section 2 covers the review of diffusion models and previous works. Preliminary information of the proposed method is introduced in Section 3. Section 4 is dedicated to the formal definition of the problem and the details of the proposed method. Details of the experiments and the results of the evaluation of the proposed method are reported in Section 5, and finally Section 6 concludes the paper.

2. Diffusion models and related works

Diffusion models are used to simulate the spreading process in real world. Some popular diffusion models are discussed in this section. In addition, a review of well-known methods for IM problem is presented in this section. A network is modeled as a graph where the nodes represent the network users, and the relationships between the users are shown by edges.

2.1. Diffusion models

Diffusion models are used to simulate the spreading process in real world. Diffusion models are generally divided into threshold (Granovetter, 1978; Kempe et al., 2003), cascading (Goldenberg, Libai & Muller, 2001; Kempe et al., 2003), and epidemic (Buscarino, Fortuna, Frasca & Latora, 2008; Peng, Yu & Yang, 2014) models. Independent cascading (IC) model (Goldenberg et al., 2001) is one of the most popular models widely used in the literature. In IC model, each user can be in either active or inactive modes. In order to examine the influentiality of a seed set S, the nodes placed in S are initially set as active users and all other users are considered as inactive. At each time step t, each user activated in step t-1 has a single chance to activate each of its inactive neighbors with activation probability p. This process continues until no new user becomes active in a time step. At the end, the number of activated users during the process is considered as the influence spread of S. In order to obtain statistically significant results, it is repeated many times, and the average number of activated users is considered as influentiality of S.

2.2. Related works

The existing research studies on IM problem can be classified into two main categories. The first category identify influentiality of users and rank them. In most of these studies, according to the network structure and users' position in the network, a centrality measure is used to measure the influentiality of the users. Degree centrality (Freeman, 1978), betweenness (Freeman, 1977), closeness (Sabidussi, 1966), k-shell (Kitsak et al., 2010), Page Rank (Brin & Page, 2012), neighborhood diversity (Zareie et al., 2019b), and hierarchical k-shell (Zareie & Sheikhahmadi, 2018) are some well-known centrality measures used for this purpose. Although classical centrality measures that are simple to computing, can determine node vitality (Davoudi & Chatterjee, 2018; Kitsak et al., 2010; Meo, Musial-Gabrys, Rosaci, Sarne & Aroyo, 2017; Yuan, Sun & Li, 2017; Zareie & Sheikhahmadi, 2018), they do not often result in high levels of accuracy in correctly identifying top-k most influential nodes. A number of measures have been proposed to optimally select seed nodes maximizing their collective influence range. One may improve further the performance, by considering not only influentiality of nodes, but also distance between them on the network (Bao et al., 2017). In order to obtain maximum collective influentiality for k selected nodes, they should be influential nodes and at the same time located in different parts of the network, i.e. distance between them is maximized (Zareie et al., 2018). This guarantees maximizing reachability of larger part of the network when the seed nodes are first activated. The second category of IM methods consider this and formulate the problem as an optimization problem. The optimization-based methods are classified into three groups, which are discussed in the following.

2.2.1. Greedy methods

In greedy method (Kempe et al., 2003), identification of a seed set S containing k most influential users is iterated in k steps and diffusion models are applied to estimate influentiality of the set at each iteration. First, S is considered as an empty set. Then, the most influential user is identified and added to S as the first member. In the second step, the influentiality of $S + \{v\}$ is determined for every $v \in V - - S$ and the user whose union with S has the highest influence, is added to S as the new member. By iterating this process, k nodes are added to S. The selected set by this method has an acceptable influence spread, but given the high time complexity, applying this method is time consuming in large networks. If the diffusion model used for numerical simulation is sub-modular, one can used methods like Cost efficient lazy forward (CELF) (Leskovec et al., 2007) to reduce the time complexity of the greedy method. Although this method is much faster than the greedy one, it still suffers from a high time overload. CELF++ (Goyal, Lu & Lakshmanan, 2011) is an improvement over the CELF, which improves its time complexity up to 35 to 55. Likewise, various methods (Heidari, Asadpour & Faili, 2015; Wang et al., 2010) have been proposed to improve the greedy method.

2.2.2. Heuristic methods

In order to improve the time complexity of the optimization process of IM problem, a series of heuristic methods have been presented for selecting the near-optimal set of influential users. In these methods, a score is assigned to each user by using a recursive approach, and the users with the highest score are selected as seeds. In Chen, Wang and Yang (2009), two methods, named single discount and double discount, were presented for selecting the seed set. In these methods, the degree of each node is initially considered as indicator of its influentiality, and selection of seeds is iterated in k steps. At each step, the node with the highest influentiality is added to the seed set as a new member. If node v is added to the set as a new member, the influentiality of its neighbors is reduced, and they will subsequently have less chance to be selected as the seed. Degree punishment method (Wang, Su, Zhao & Yi, 2016) also tries to select the seeds by the same approach. In this method, if node v is selected as a seed, the influentiality of its neighbors and second-order neighbors is

punished, and their chance to be selected as seed is reduced. This strategy leads to less overlap and more dispersion of the seeds in the network. In degree distance method (Sheikhahmadi et al., 2015), identification of influential users is similarly repeated in k steps, where at each step, the node with the highest degree is considered as a candidate to be added to the seed set as a new member. If there is an acceptable distance between the candidate and each member of the seed set, it is added to the set as a new member; otherwise the node with the next highest degree is nominated as the candidate. In Guo, Lin, Guo and Liu (2016), a distance-based graph coloring method is presented to identify the seeds with proper distance to each other. This method tries to classify the nodes so that the distance between each pair of the nodes in each group is higher than a threshold. In Bao et al. (2017), a heuristic clustering method was presented, where the similarity between each pair of nodes is first determined, and nodes are clustered in k different clusters. Then, the most influential node in each cluster is selected as members of the seed. IM problem is modeled as a multi-objective optimization problem and a heuristic model is proposed to solve the problem (Zareie et al., 2018).

In spite of low time complexity, the heuristic methods suffer from low accuracy in selecting the near-optimal set of influential nodes (Cui et al., 2018), which is mainly due to being trapped in a local optimal solution.

2.2.3. Meta-heuristic methods

In these methods, by defining a fitness function, IM problem is modeled as an optimization problem, and methods like evolutionary optimization algorithms are applied to solve it. In Jiang et al. (2011), the influentiality of the seed set S is defined as Expected Diffusion Value (*EDV*), as:

$$EDV(S) = k + \sum_{u \in N_s^{(1)} \setminus S} \left(1 - (1-p)^{t(u)}\right) \quad (1)$$

where $N_s^{(1)}$ is the neighbors of the members of S and $N_s^{(1)} \setminus S$ denotes the set which are members of $N_s^{(1)}$ but not members of S. p shows the activation probability. In spreading process, each node $u \in N_s^{(1)} \setminus S$ will not be activated with probability $(1-p)^{t(u)}$, where $t(u)$ is the number of neighbors of node u which are members of S, it will be accordingly activated with probability $1-(1-p)^{t(u)}$. Thus, the size of the seed set, k, as well as the number of neighbors being activated in the spreading process, is considered as in Eq. (1). The simulated annealing algorithm was used to find S with the aim of maximizing *EDV*. In Gong et al. (2016), an improvement of *EDV* is introduced, and Particle Swarm Optimization (PSO) algorithm is used to maximize the objective function. Cui et al. (2018) proposed a method find S by using differential evolution algorithm with the aim of maximizing the *EDV* function. Replica symmetrically mean-field theory is used to solve IM problem in Sun (2016). Apart from nodes' influence, budget constraint can also be taken into account to define the IM as an optimization problem (Borrero, Prokopyev & Krokhmal, 2018; Eshghi et al., 2018; Preciado, Zargham, Enyioha, Jadbabaie & Pappas, 2013).

The overlap between the members of the seed has been neglected in all of the above mentioned meta-heuristic methods, and the selected seeds may cover limited numbers of nodes with high activation probabilities. Therefore, in this paper, a fitness function is first defined, and then by using the gray wolf optimization algorithm a method is presented to maximize the fitness function.

3. Preliminary information

In the proposed algorithm, by applying the concept of entropy a fitness function is first defined and IM problem is modeled as an optimization problem. The problem is then solved using gray wolf optimization algorithm. The concept of entropy was firstly introduced by Shannon in 1948 for determining the amount of information in an event (Jaynes, 1957). Based on the Shannon entropy, if X is a set of possible events $x_1,\ldots, x_n$, and p_i is the probability of occurrence of the event x_i such that $\sum_{i=1}^{n} p_i = 1$, then the entropy of X is computed using Eq. (2).

$$H(X) = -\sum_{i=1}^{n} p_i \times ln(p_i) \quad (2)$$

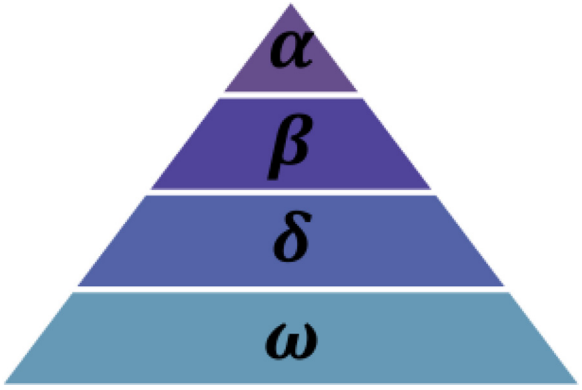

Fig. 1. Social dominant hierarchy of gray wolves group (Mirjalili et al., 2014).

Gray wolf optimization (GWO) Algorithm (Mirjalili et al., 2014) is a population-based evolutionary algorithm inspired the hunting behavior of gray wolves. As shown in Fig. 1, these wolves follow a strict social dominant hierarchy, and are divided into four categories in the society: Alpha (α), Beta (β), Delta (δ), and Omega (ω) wolves.

Alpha, Beta, and Delta wolves are responsible for managing the attack in the hunt process, and Omega wolves do not have a specific performance for the group, and they mostly play the role of scapegoat. According to the lifestyle and hunting process of these animals, GWO algorithm is modeled as follows.

First, a set of solutions is randomly generated. Each solution is assigned to a wolf indicating its position. The wolf which has the best fitness is considered as the Alpha, and the wolves having the second and third best fitness are considered as Beta and Delta wolves, respectively. The other wolves are considered as Omega wolves. In this algorithm, the problem space is searched according to the position of Alpha, Beta, and Delta wolves to find the position of the prey (optimal solution). In other words, these three wolves estimate the position of the prey, and the omega wolves update their position based on the position of these wolves to find a closer position to the prey. The algorithm attempts to find the optimal solution during iterations. In each iteration, Omega wolves try to improve their position according to the position of the other dominant wolves. In each iteration t, Omega wolf i determines its new position for iteration $t+1$, $X_i(t+1)$, as Eq. (3).

$$\vec{X}_i(t+1) = \frac{\vec{Y}_1 + \vec{Y}_2 + \vec{Y}_3}{3} \quad (3)$$

That is to say each omega wolf i seeks to find a better position based on the three values $\vec{Y}_1$, $\vec{Y}_2$, and $\vec{Y}_3$, where the value $\vec{Y}_1$ is calculated according to the current position of the wolf and the position of alpha wolf, as Eq. (4).

$$\vec{Y}_1 = \vec{X}_\alpha - \vec{A}_1.\vec{D}_\alpha \text{ where } \vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}_i(t)\right| \quad (4)$$

where t represents the current iteration, and $\vec{X}_i(t)$ expresses the position of Omega wolf i in this iteration. $\vec{X}_\alpha$ indicates the positions of Alpha.

Using Eqs. (5) and (6) the values of $\vec{Y}_2$ and $\vec{Y}_3$ are calculated according to the current position of the wolf and that of beta wolf

Algorithm 1 Pseudo-code of GWO Algorithm.

```
Initialize the random position of each gray wolf as X_i(i = 1, ..., n)
Initialize a, A, and C
Calculate the fitness value for each wolf
Set X_α, X_β, and X_δ as best, the second best, and the third best wolves, respectively
Consider the other wolves as Omega wolf
Set t = 0
While (t < max_t)
    For each Omega wolf X_i
        Update the position of X_i using Eq. (3)
    End for
    Update a, A, and C using Eqs. (6) to (8)
    Calculate the fitness value for each Omega wolf
    Update X_α, X_β and X_δ
    t = t + 1
End while
Return X_α as selected solution
```

and delta wolf, respectively.

$$\vec{Y}_2 = \vec{X}_\beta - \vec{A}_2.\vec{D}_\beta \text{ where } \vec{D}_\beta = \left|\vec{C}_2.\vec{X}_\beta - \vec{X}_i(t)\right| \tag{5}$$

$$\vec{Y}_3 = \vec{X}_\delta - \vec{A}_3.\vec{D}_\delta \text{ where } \vec{D}_\delta = \left|\vec{C}_3.\vec{X}_\delta - \vec{X}_i(t)\right| \tag{6}$$

where $\vec{X}_\beta$ and $\vec{X}_\delta$ are the position of beta and delta wolves, respectively. A and C are coefficient vectors, which are calculated by Eqs. (7) and (8).

$$\vec{A} = 2a.\vec{r}_1 - a \tag{7}$$

$$\vec{C} = 2.\vec{r}_2 \tag{8}$$

In Eqs. (7) and (8), $\vec{r}_1$and $\vec{r}_2$ are random values in the range [0, 1], and a is a control parameter that approaches linearly from 2 to 0 during the iterations. Eq. (9) is used to calculate the value of a in iteration t.

$$a = 2 - 2.\left(\frac{t}{max\_t}\right) \tag{9}$$

where max_t shows the number of iterations over the algorithm.

Over the course of the iterations, the contradiction among the position of the wolves reduces, and the algorithm converges. At the end of the algorithm, the best solution, which is the Alpha wolf, is returned as the optimal solution. The pseudo-code of GWO algorithm is shown in Algorithm 1, in which n represents the size of the population, i.e. the number of wolves.

4. The proposed IM algorithm

A social network can be represented as an undirected graph $G = (V, E)$ where $V = \{v_1, v_2, \ldots, v_{|V|}\}$ shows the network users, and E shows the relationship between them. Edge $e_{ij} \in E$ represents the edge between two nodes v_i and v_j; these two nodes are called neighbors. $N_i \subset V$ is used to represent the neighbors of node v_i, and $d_i = |N_i|$ represents the degree of v_i. $N_i^{(2)}$ shows the second-order neighbors, i.e. neighbor of neighbor, of v_i. The goal of IM problem is to identify a set S with k members to initiate the spreading process so that the spread is maximized, i.e. the number of activated users is maximized. In the rest of the paper, S' shows the set of nodes located in the neighborhood or second-order neighborhood of seed set S.

4.1. Fitness function

The proposed algorithm select nodes as members of the seed set, such that (*i*) each node has high influentiality, and (*ii*) the selected set covers broad parts of the network as much as possible. This guarantees maximum propagation. During the propagation process, the probability that node $v_j \in S'$ is activated, i.e. receives the message, is calculated as:

$$I(v_j) = \sum_{e_{ij} \in E \text{ and } v_i \in S} p_{ij} + \sum_{e_{ik} \in E \text{ and } e_{kj} \in E \text{ and } v_i \in S} \left(p_{ik} \cdot p_{kj} \right) \tag{10}$$

where p_{ij} shows the probability of propagation from v_i to v_j. The first section of Eq. (10) calculates the probability that node v_j receives the message of its neighbors, which are members of S; it is considered 0 if node v_j has no neighbors in S. The second section of the equation calculates the probability that node v_j receives the message from its second-order neighbors that are also members of S. These two parts are summed due to the probability rules.

Different nodes do not have the same impact during the propagation process. If the nodes with high degree receive the message, the message may be spread to further parts of the network. Therefore, the worthiness of each node $v_j \in S'$ is calculated by:

$$w(v_j) = I(v_j).\, d_j \tag{11}$$

Therefore, in the proposed algorithm, called Gray Wolf based Influence Maximization (GWIM), Eq. (12) can be considered as the fitness function.

$$W(S) = \sum_{v_j \in S'} w(v_j) \tag{12}$$

Applying summation operator in the fitness function might not be efficient as it cannot ensure the minimum overlap between the seeds. In GWIM, we seek to find the seeds, such that (*i*) the number of nodes in S' is maximized, and (*ii*) all of these nodes have high worthiness. In the case of using summation operator, a composition of a limited number of worthy nodes and significant number of unworthy ones in S' may maximize the value of the fitness function (12). In this case, failure in propagation of the message towards the worthy nodes can lead to dramatic decrease in the influentiality of the selected seeds. Entropy is used in the fitness function to tackle this issue, which indeed determines how uniform distribution of worthiness of nodes in S' can determine influence spread of S more accurately. On the other hand, the value of entropy rises as the size of S' increases, and thus a proper normalization needs to be considered. The entropy of the worthiness as the fitness value of S, is defined as:

$$f(S) = H(S) = -\sum_{v_j \in S'} \frac{w(v_j)}{W(S)} \times ln\left(\frac{w(v_j)}{W(S)}\right) \tag{13}$$

where $W(S)$ is the sum of the worthiness of the nodes in $S'$and is calculated using Eq. (12).

Algorithm 2 pseudo-code of GWIM .

```
01 Input: Undirected Graph G=(V,E), the seed set size k, the population size n, the number of iterations max_t
02 Output: S // a set with k members as initial seed set
03 Begin Algorithm
04 Initial a, A, and C and |V'|={v_i ∈ V | d_i⟩1}
05     Initial n position X_1 to X_n randomly and determine S_i corresponded to each X_i
06     Calculate the fitness for each S_i (i=1, 2, ..., n)
07     According to the fitness values, select the best solution as X_α, the second best solution as X_β, the third best solution as X_δ, and the
        others as Omega solutions
08     Set t=0
09     While (t <max_t)
10         For each Omega wolf i
11             Update the position of X_i and determine corresponded S_i
12         End for
13         Update a, A, and C
14         Calculate the fitness for each S_i (i=1, 2, ..., n)
15         Update X_α, X_β and X_δ and call random position if necessary
16         t=t+1
17     End while
18     Return X_α
19 End Algorithm
```

4.2. Proposed algorithm

According to Sheikhahmadi et al. (2017), the nodes with degree 1 have a very low chance to be selected as seeds. In many real social networks, there are large portion of nodes with only one neighbors. Therefore, in order to reduce the time complexity of GWIM algorithm, only nodes with degree higher than 1 are considered as possible seeds; let's denote such nodes as $V' = \{v'_1, v'_2, \ldots, v'_{|V'|}\}$. During the proposed algorithm, each wolf (solution) has two property: position and corresponding seed set. The position of wolf i is shown as a vector X_i with $|V'|$ elements, where the j-th indicates the chance of node v'_j to be selected as a seed. The corresponding seed set of wolf i is shown as S_i, which contains k nodes with the highest values in X_i.

Algorithm 2 shows the pseudo-code of GWIM algorithm. In line 05 of the algorithm, n primary solutions are generated randomly. Random position function is used to generate each random solution. In line 06, the fitness value of the solutions is computed using (12). In line 07, according to the fitness value of the solutions, Alpha, Beta, and Delta wolves are determined. In lines 9–17, the operations are iterated in max_t times to maximize the fitness function of the wolves. In each iteration, by using Update position function, the position of Omega wolves is updated, and given to the updated positions. Seed set S_i is determined for each wolf i in lines 10–12. In line 14, the fitness value is calculated for each S_i, and according to the fitness value, the best solutions are considered as new Alpha, Beta, and Delta wolves. In line 15, the position of Beta and Delta wolves are randomly regenerated, if X_β is equal to X_α or X_δ is equal to X_β. This helps avoiding being trapped in local optimal solution. The functions used in Algorithm 1 are further described below.

4.2.1. Generation of primary population

In GWIM algorithm, Random position function is used to generate a primary solution i (primary position X_i of wolf i and its corresponding seed set S_i). The pseudo-code of this function is shown in Algorithm 3. In this algorithm, a random value X_{ij} is given to each node $v_j \in V'$, as its chance of being selected as a seed, which is proportional to their degree (line 04). k nodes corresponding to the k highest entries of X_i are chosen as S_i members in lines 08 to 11.

4.2.2. Updating Wolves' position

In GWIM algorithm, according to the position of Alpha, Beta and Delta wolves, the position of each Omega wolf is updated with the hope of achieving a better position. Update position function is defined for this purpose. The pseudo-code of this function is shown in Algorithm 4. In this algorithm, in order to update the chance of node v_j in wolf i, X_{ij}, the chance of this node in Alpha wolf, $X_{\alpha j}$, Beta wolf, $X_{\beta j}$, and Delta wolf, $X_{\delta j}$, is used. Therefore, the values of Y_1, Y_2,and Y_3 are calculated for node v_j, and the value of $X_{ij}(t+1)$ is accordingly calculated. By calculating the value of $X_{ij}(t+1)$ for each node $v_j \in V'$ the position of wolf i for iteration $t+1$, $X_i(t+1)$, is updated. In Algorithm 4, $Abs(a)$ shows the absolute value of a.

Algorithm 3 Random position function.

```
01 Input: Undirected Graph G=(V,E), V', the seed set size k.
02 Output: X_i, the position of wolf i, and S_i, corresponded seed set.
03     For j=1 to |V'|
04         r=rand (1...)
05         X_ij=r/max(d)
06     End for
07     Set S_i={}
08     For i=1 to k
09         Find the next maximum X_ij
10         Add v_j to S_i
11     End for
12     Return X_i and S_i
13 End Algorithm
```

For example, consider a network in which nodes 1, 4, 5, 7, 15, 17, 21, 29, 30, 40 have degree greater than 1, and we fix $k=3$. Here, $V'=\{1, 4, 5, 7, 15, 17, 21, 29, 30, 40\}$ and $|V'|=10$, and thus the length of each position is 10. Let's suppose the position vector X_i assigned to wolf i as:

1	4	5	7	15	17	21	29	30	40
0.336	0.215	0.101	0.156	0.546	0.489	0.112	0.356	0.686	0.125

Each X_{ij} in the position shows the chance of jth node in $V'$to be selected as the seed. For example, the chance of node 1 is 0.336, while it is 0.215 for node 4. k nodes with the highest chances are selected as the corresponding solution that is$S_i=\{15, 17, 30\}$. Consider that max_$t=10$ and we want to update the position of wolf i in the sixth iteration, i.e. $t=6$. a is calculated as $a = 2-2\times(\frac{6}{10}) = 0.8$. $\vec{A}$ and $\vec{C}$ are two matrices with 3 rows and $|V'|$ columns. A random value in the range [0, 1] is generated for $\vec{r}_{11}$. Suppose $\vec{r}_{11}=0.25$. Thus, the first entry of vector $\vec{A}_1$ (that is $\vec{A}_{11}$), is calculated as $2\times0.8\times0.25-0.8=-0.4$. This process is repeated to update all entries of $\vec{A}$. Matrix $\vec{C}$ is updated using a similar procedure based on random values in $\vec{r}_2$ (see Eq. (8)). Let's suppose that the position and corresponding solutions for alpha, beta and delta wolves are as follow:

Algorithm 4 Update position function.

Input: A, C, V', $X_i(t)$, X_α, X_β and X_δ
Output: $X_i(t+1)$ //updated X_i
  For $j = 1$ **to** $|V'|$
    $D_{\alpha j} = Abs(C_1.X_{\alpha j} - X_{ij}(t))$ (X_{ij} is the jth entry of Xi)
    $Y_1 = X_{\alpha j} - A_1.D_{\alpha j}$
    $D_{\beta j} = Abs(C_2.X_{\beta j} - X_{ij}(t))$
    $Y_2 = X_{\beta j} - A_2.D_{\beta j}$
    $D_{\delta j} = Abs(C_3.X_{\delta j} - X_{ij}(t))$
    $Y_3 = X_{\delta j} - A_3.D_{\delta j}$
    $X_{ij}(t+1) = \frac{Y_1+Y_2+Y_3}{3}$
  End for
  Return $X_i(t+1)$
End Algorithm

Wolf		Position										Solution
Alpha	$\vec{X}_\alpha$	1	4	5	7	15	17	21	29	30	40	{1, 15, 17}
		0.412	0.185	0.102	0.125	0.564	0.465	0.102	0.359	0.325	0.119	
Beta	$\vec{X}_\beta$	1	4	5	7	15	17	21	29	30	40	{1, 15, 30}
		0.512	0.215	0.115	0.112	0.602	0.325	0.215	0.289	0.402	0.190	
Delta	$\vec{X}_\delta$	1	4	5	7	15	17	21	29	30	40	{5, 29, 30}
		0.325	0.300	0.365	0.075	0.300	0.352	0.236	0.490	0.458	0.250	

The value of $\vec{Y}_{11}$, $\vec{Y}_{21}$, and $\vec{Y}_{31}$ are then calculated to update the first entry of X_i (that is X_{i1}). Let's suppose that $\vec{C}_{11} = 0.3$, $\vec{C}_{21} = 0.4$, $\vec{C}_{31} = 0.5$, $\vec{A}_{11} = -0.4$, $\vec{A}_{21} = -0.3$, $\vec{A}_{31} = 0.1$. We then have:

$$\vec{D}_{\alpha 1} = \left|\vec{C}_{11}.\vec{X}_{\alpha 1} - \vec{X}_{i1}(5)\right| = |0.3 \times 0.412 - 0.336| = 0.212$$

$$\vec{Y}_{11} = \vec{X}_{\alpha 1} - \vec{A}_{11}.\vec{D}_{\alpha 1} = 0.412 - (-0.4) * 0.212 = 0.497$$

$$\vec{D}_{\beta 1} = \left|\vec{C}_{21}.\vec{X}_{\beta 1} - \vec{X}_{i1}(5)\right| = |0.4 \times 0.512 - 0.336| = 0.131$$

$$\vec{Y}_{21} = \vec{X}_{\beta 1} - \vec{A}_{21}.\vec{D}_{\beta 1} = 0.512 - (-0.3) * 0.131 = 0.551$$

$$\vec{D}_{\delta 1} = \left|\vec{C}_{31}.\vec{X}_{\delta 1} - \vec{X}_{i1}(5)\right| = |0.5 \times 0.325 - 0.336| = 0.174$$

$$\vec{Y}_{31} = \vec{X}_{\delta 1} - \vec{A}_{31}.\vec{D}_{\delta 1} = 0.325 - (0.1) * 0.174 = 0.308$$

Thus, the updated first entry of X_i in the 6*th* iteration is calculated as follow:

$$\vec{X}_{i1}(6) = \frac{\vec{Y}_{11} + \vec{Y}_{21} + \vec{Y}_{31}}{3} = \frac{0.497 + 0.551 + 0.308}{3} \cong 0.452$$

This process is repeated for each $\vec{X}_{ij}$ ($j \in 1 \ldots |V'|$), and the position of wolf i is updated in the 6*th* iteration. This calculations are repeated to update the position of all wolves in each iteration.

4.3. Computational complexity analysis

The proposed algorithm has two main steps: the initialization step (lines 4 to 8) and the main step (lines 9 to 17). Complexity of line 4 is $O(|V|)$ due to assessing of all nodes to determine $|V'|$. Having the complexity $O(|V'| + k.|V'|)$, the random position function is called for n times in line 5. Thus, the complexity of line 5 is $O(n.|V'| + n.k.|V'|)$. The fitness value is calculated for n solutions in line 6 using Eq. (12), with the complexity $O(k.d.\ d^{(2)})$, where d and $\langle d^{(2)} \rangle$ are the average degree and second-order degree of nodes. Thus, complexity of line 6 is $O(n.k.d.\ d^{(2)})$. The identification of three best solutions can be done with complexity $O(n)$ in line 7. Thus, the computational complexity of the initialization step is $O(|V| + n.|V'| + n.k.|V'| + n.k.d.\ \langle d^{(2)} \rangle + n)$, which can be reduced to $O(|V| + n.|V'| + n.k.|V'| + n.k.d.\ d^{(2)})$.

In lines 10–12, the position of each omega wolf is updated using the Update position function, which has the complexity $O(|V'|)$. Thus, the complexity of lines 10–12 is $O(n.|V'|)$. As it was mentioned, the complexity of the calculation of n solutions in line 14 and the identification of three best solutions in line 15 are $O(n.k.d.\ d^{(2)})$ and $O(n)$, respectively. The complexity of the main step of the proposed algorithm is $O(max_t.(n.|V'| + n.k.d.\ d^{(2)} + n))$, which can be reduced to $O(max_t.(n.|V'| + n.k.d.\ d^{(2)}))$. The overall time complexity of the proposed algorithm is $O(|V| + n.|V'| + n.k.|V'| + n.k.d.\ d^{(2)} + max\_t.\ n.|V'| + max\_t.n.k.d.\ d^{(2)})$, which can be reduced to $O(|V| + max\_t.\ n.|V'| + max\_t.n.k.d.\ d^{(2)})$.

5. Experiments and evaluation of the proposed method

For the experiments, three real-world networks are used and their details are shown in Table 1. In this table, p is the activation probability used in IC model. It is set according to the sparsity of the graphs and is calculated based on the average of the degree and second-order degrees of nodes in the network.

In order to assess performance of the proposed method, the results are compared with a number of state-of-the-art influence maximization methods, including:

- Centrality measure:
  - Degree centrality (Freeman, 1978)
  - k-shell (Kitsak et al., 2010) (Kitsak et al., 2010)
  - Page Rank (Brin & Page, 2012)
  - Betweenness centrality (BC) (Freeman, 1977)
- Greedy method:

Table 1
Details of the real-world networks used in the experiments; |V| is the number of nodes, |E| is the number of edges, ⟨d⟩ is the average degree of nodes, and p is activation probability used in independent cascade diffusion model.

Network	$\|V\|$	$\|E\|$	$\langle d \rangle$	p
Hamsterster full (HAM)	2426	16,631	13.711	0.03
Pretty Good Privacy (PGP)	10,680	24,316	4.5536	0.06
Astro (AST)	18,771	198,050	21.102	0.02

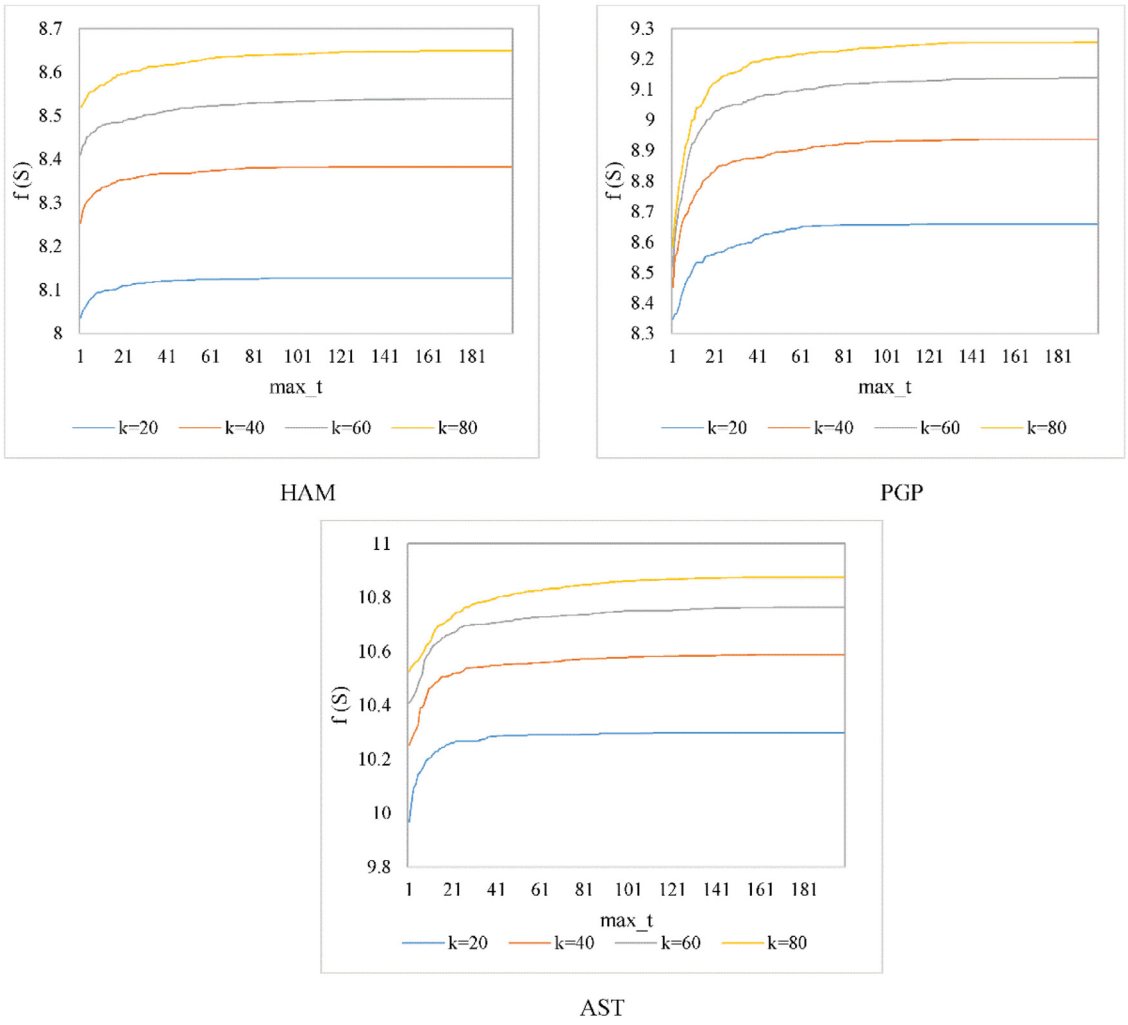

Fig. 2. The fitness function as a function of max_t in different networks.

- CELF++ (Goyal et al., 2011)

○ Heuristic methods:
- double discount (DD) (Chen et al., 2009)
- heuristic clustering (HC) (Bao et al., 2017)

○ Meta-heuristic methods:
- Degree Descending Search Strategy (DDSE) (Cui et al., 2018)
- Simulated annealing EDV (SADV) (Jiang et al., 2011)

The experiments are repeated 20 times, and the results show averages over these runs.

5.1. Parameters of GWO

In the first experiment, the effects of parameters of GWO on the final value of the fitness function is examined, and the optimal value for the population size n and the number of iteration max_t is achieved. To analyze the impact of max_t, population size n is set 50, and the value of max_t is varied from 1 to 200. The results of this experiment on various networks for $k=20$, 40, 60, 80 are shown in Fig. 2. As can be seen, as max_t exceeds 100, the value of the fitness function does not increase significantly; hence, in the next experiments, the value of max_t is set to 100.

In the next experiment, the effect of population size n on the value of fitness is investigated. To do so, the maximum iteration max_t is set at 100, and the value of the obtained fitness in different iterations for population sizes of 10, 30, 50, 70, 90 is reported. The results, see Fig. 3, show that the value of the fitness function increases by increasing the population size. However, no significant increase is obtained when the population size takes values higher than 50, and thus the population size is fixed at 50 for the following experiments.

5.2. Convergence speed

The convergence speed of the proposed method is investigated in this section. To this end so, the movement of the wolves during the optimization process is studied. For this, the Euclidian distance between the positions of each wolf i in two sequential iterations t and $t+1$ is calculated using:

$$mov(i,t,t+1)=\sqrt{\sum_{j=1}^{|v|}\left(\vec{X}_{ij}(t+1)-\vec{X}_{ij}(t)\right)^2} \tag{14}$$

where $\vec{X}_{ij}(t)$shows the value of jth entry of the position of wolf i in iteration t. The average movement of all wolves in two sequential iterations t and $t+1$ is calculated as:

$$AM(t,t+1)=\frac{\sum_{i=1}^{n} mov(i,t,t+1)}{n} \tag{15}$$

where n is the size of population

In this experiment, the size of seed set is considered as $k=40$. Fig. 4 illustrates the results. As can be seen, the average movement increases during the initial iterations. As the number of iterations

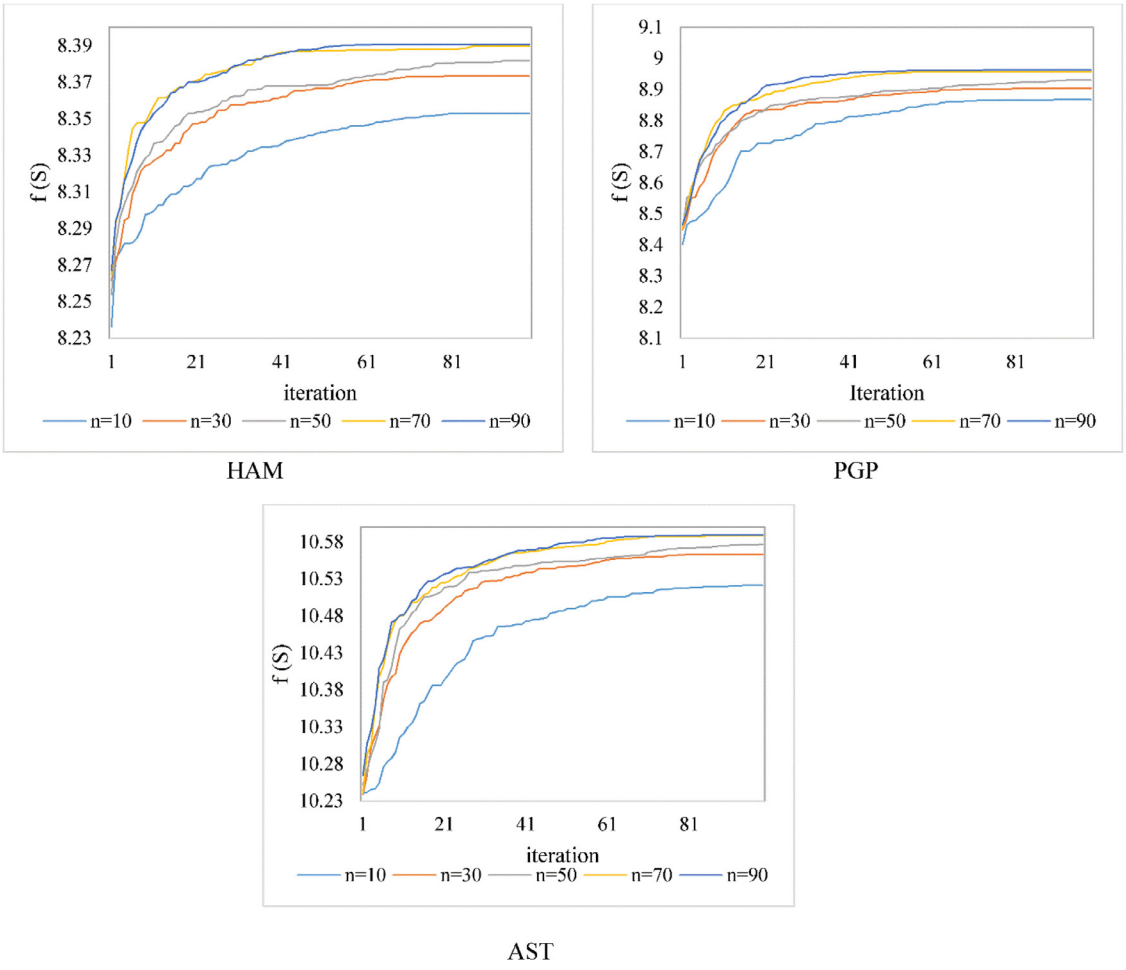

Fig. 3. The fitness function as a function of iterations for different population size n.

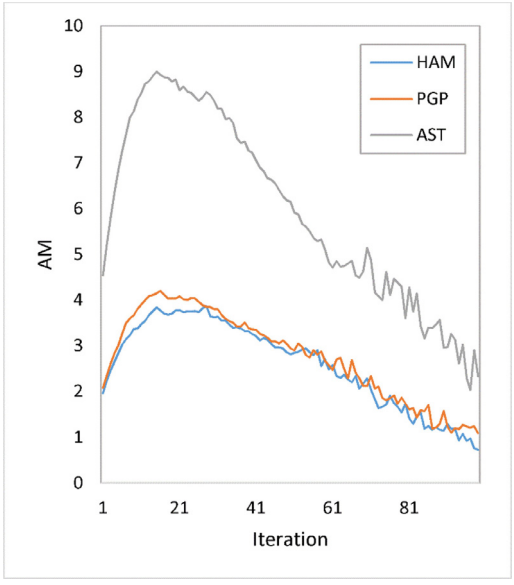

Fig. 4. The average movement of wolves as a function of the iterations.

increases, the average movement of the wolf decreases, and the algorithm converges. It is also seen that in spite of oscillatory behavior in the final iterations, which is caused by line 15 in Algorithm 2 to avoid being trapped in local optimal solution, the proposed algorithm experiences a convergence trend in the final iterations.

5.3. Evaluation of influentiality of the seeds obtained by the proposed method

We next evaluate the performance of GWIM with other algorithms by comparing the influentiality of the seeds selected by these algorithms. For this, the size of seed set is varied from 10 to 80. IC model is applied to evaluate the influentiality of each seed set offered by different algorithms. In order to improve the confidence of the numerical results, IC model is repeated 50 times in the experiments of this section. The obtained results shown in Fig. 5 express that GWIM and CELF++ methods explore the search space better than other methods, and results in a highest influentiality for the same seed size. GWIM provides close results to CELF++, and especially, for greater size of seed set, GWIM explores the search space better than CELF++, and outperforms it. Considering the distance between seeds plays more important role as seed set size increases, and so selected seed set by GWIM propose higher influentiality than other ones. Expecting, as the seed size increases, the influentiality of IM algorithms also increases. DDSE that is another method that uses an optimization process is the third top-performer after GWIM and CELF++.

In the next experiment, we examine the effect of varying the activation probability of IC model on the performance of

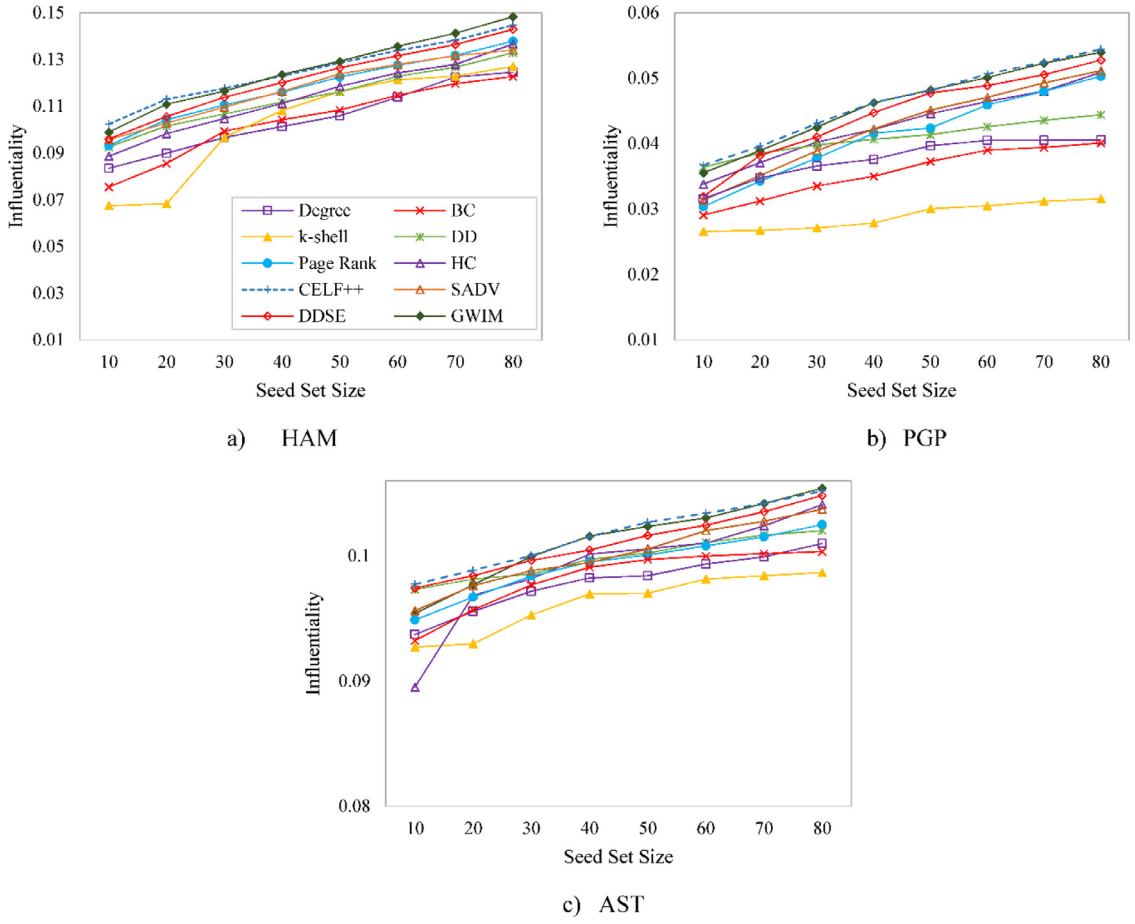

Fig. 5. The influentiality (the number of activated nodes divided by the network size) as a function of size of the seed set. The algorithms including the proposed algorithm (GWIM) are applied on three networks.

the algorithms. To do so, the value of k is set to 40, and the value of the activation probability is varied from 0.01 to 0.1 with interval 0.01. The results of this experiment are shown in Fig. 6, where by increasing p, the influentiality also increases. Almost for all cases, the proposed algorithm is the top-performer, and its outperformance to other algorithms is more pronounced as p increases. The worthiness of nodes increases as p increases, and applying entropy notation in the fitness function leads to select seeds with less overlap in GWIM. As a result, GWIM outperforms the other algorithms in higher activation probability.

5.4. Statistical tests

In this section, the differences in performance of the algorithms are tested for statistical significance. To make statistical significance, we conduct Friedman test analysis and use Bonferroni method to correct for multiple comparisons. The confidence score is set as $\alpha = 0.05$, which means that the differences are statically significate if the p-value is less than 0.05. The p-values are reported in Table 2. In this table, the method with the best results in each value of k is shown by '_' and is accordingly considered as control method. Boldface values indicate that the corresponding hypothesis are accepted. For each value of k, the algorithms are ascendingly ordered based on their p-values. If the ith p-value in the ordered list is lower than $\alpha/(N\text{-}i)$, where N is the number of the algorithms, then the corresponding hypothesis is rejected; otherwise it is accepted. As can be seen from the table, for the values of k as 10, 20 and 30, CELF++ has the best results and is considered as control method. For $k = 10$, some algorithms including SWIM provide close performance to CELF++. For the other values of k, GWIM obtains the best results and is considered as the control method. Furthermore, DDSE and CELF++ obtain close performance to GWIM, while CELF++ outperforms DDSE.

Method	$k=10$	$k=20$	$k=30$	$k=40$	$k=50$	$k=60$	$k=70$	$k=80$
Degree	0.007	0.000	0.000	0.000	0.000	0.000	0.000	0.000
BC	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
k-shell	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
DD	**1.000**	**0.158**	0.000	0.001	0.000	0.000	0.000	0.000
Page Rank	**0.023**	0.000	0.000	0.002	0.000	0.000	0.000	0.000
HC	0.007	0.001	0.000	0.004	0.004	0.000	0.001	0.001
SADV	**0.255**	0.005	0.001	0.010	**0.030**	**0.065**	**0.032**	0.000
CELF++	_	_	_	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
DDSE	**1.000**	**1.000**	**0.643**	**1.000**	**1.000**	**1.000**	**1.000**	**0.503**
GWIM	**1.000**	**1.000**	**1.000**	_	_	_	_	_

5.5. Time complexity

We next evaluate efficiency of the algorithms. To this end, the average execution time of the algorithms is compared (Fig. 7). GWIM is more complex than centrality-based methods, but it is more computationally efficient than other meta-heuristic methods, while also resulting in better performance in terms of the influentiality of the obtained seed set. CELF++, has the worst time complexity compared to other methods.

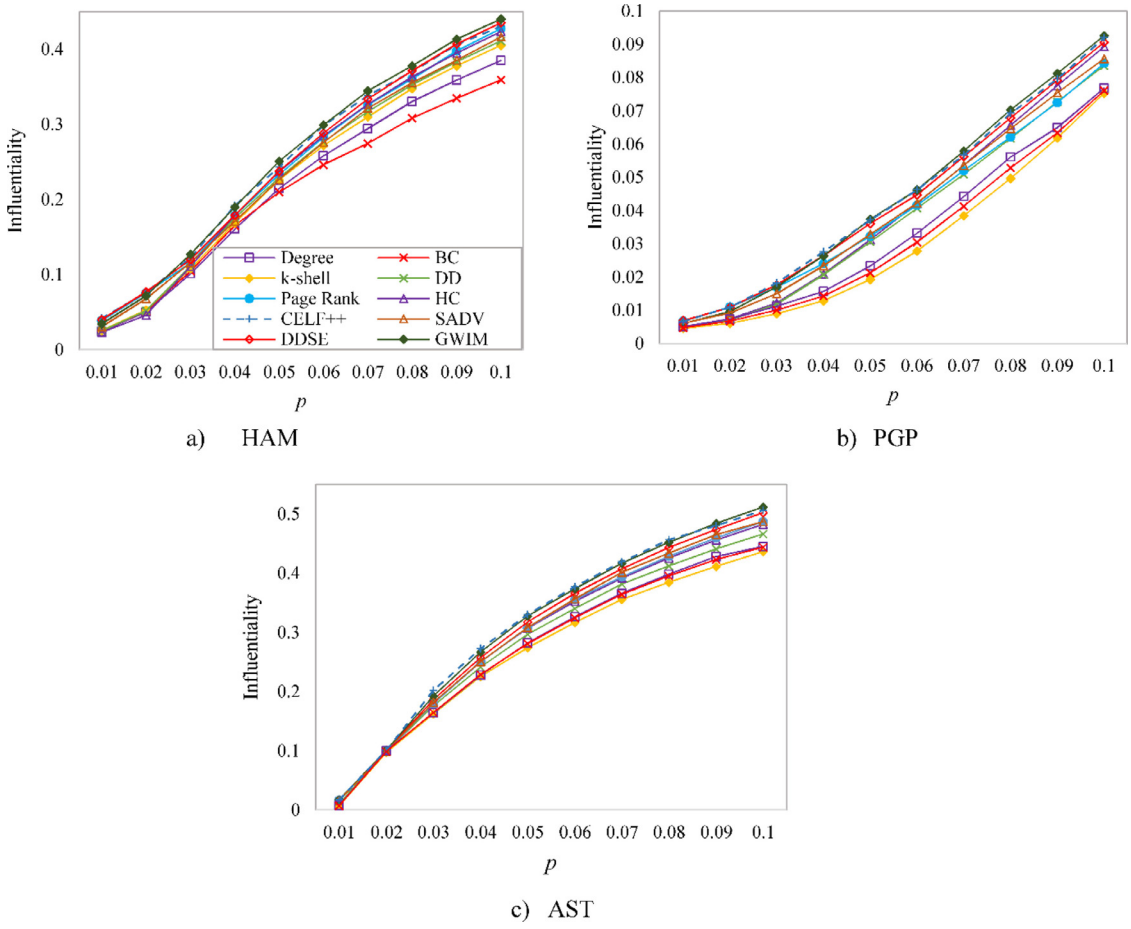

Fig. 6. The influentiality (the number of activated nodes divided by the network size) as a function of activation probability. The algorithms including the proposed algorithm (GWIM) are applied on three networks.

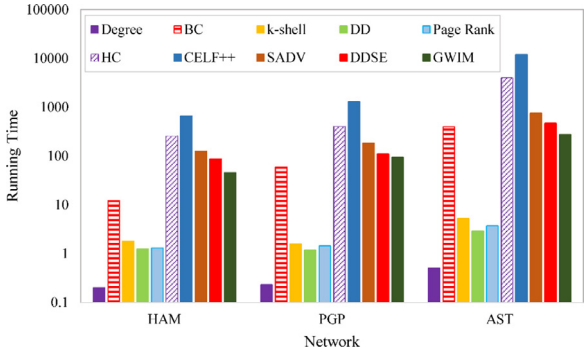

Fig. 7. The average run time of the algorithms.

6. Conclusion

Effective propagation of information in network systems has important implications in many applications, such as e-commerce and media. Under limited budget, a major challenge in viral marketing is to identify small number of influential users, called seed set. By initially passing the information to the seed set, one might expect to obtain large effect in the network, as the nodes of the seed set can influence others through their connections. This problem is referred to as influence maximization problem is the literature. The influence maximization problem can be modelled as an optimization problem with cost functions such as the influentiality of the nodes and the distance between them. In this paper, we used gray wolf optimization algorithm, as a population-based optimization method, for the influence maximization problem. Our experiments on three real-world networks showed that the proposed algorithm outperforms a number of state-of-the-art influence maximization algorithms. Also, it is not only more effective than other meta-heuristic methods, but also has less computational time. The proposed method has some limitations, which is mainly due to applying population-based optimization approach. Although it tries to escape from local minima, it can still trap in a local minima, meaning that the final solution will be a local minima rather than a global minima. However, our simulations show that it still results in better performance than state-of-the art.

Declaration of Competing Interest

The authors declared that they have no conflicts of interest to this work. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

Credit authorship contribution statement

Ahmad Zareie: Conceptualization, Methodology, Software, Data curation, Validation, Writing - original draft. **Amir Sheikhahmadi:** Conceptualization, Resources, Supervision, Project administration, Validation, Writing - review & editing. **Mahdi Jalili:** Conceptualization, Validation, Writing - review & editing.

References

Bao, Z.-. K., Liu, J.-. G., & Zhang, H.-. F. (2017). Identifying multiple influential spreaders by a heuristic clustering algorithm. *Physics Letters A, 381*(11), 976–983.

Borrero, J. S., Prokopyev, O., & Krokhmal, P. (2018). Optimization of cascading processes in arbitrary networks with stochastic interactions. *IEEE Transactions on Network Science and Engineering*.

Brin, S., & Page, L. (2012). Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks, 56*(18), 3825–3833.

Buscarino, A., Fortuna, L., Frasca, M., & Latora, V. (2008). Disease spreading in populations of moving agents. *EPL, 82*(3), 38002.

Chen, W., Wang, Y., & Yang, S. (2009). Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Chevalier, J. A., & Mayzlin, D. (2006). The effect of word of mouth on sales: Online book reviews. *Journal of Marketing Research, 43*(3), 345–354.

Cui, L., Hu, H., Yu, S., Yan, Q., Ming, Z., Wen, Z., et al. (2018). DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks. *Journal of Network and Computer Applications, 103*, 119–130.

Davoudi, A., & Chatterjee, M. (2018). Social trust model for rating prediction in recommender systems: Effects of similarity, centrality, and social ties. *Online Social Networks and Media, 7*, 1–11.

Eshghi, S., Preciado, V., Sarkar, S., Venkatesh, S., Zhao, Q., D'Souza, R., et al. (2018). Spread, then target, and advertise in waves: Optimal budget allocation across advertising channels. In *IEEE Transactions on Network Science and Engineering* (p. 1). doi:10.1109/TNSE.2018.2873281.

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry, 40*(1), 35–41.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks, 1*(3), 215–239.

Goldenberg, J., Libai, B., & Muller, E. (2001). 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*(3), 1–18.

Gong, M., Yan, J., Shen, B., Ma, L., & Cai, Q. (2016). Influence maximization in social networks based on discrete particle swarm optimization. *Information Sciences, 367*, 600–614.

Goyal, A., Lu, W., & Lakshmanan, L. V. (2011). Celf++: Optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th international conference companion on World wide web*.

Granovetter, M. (1978). Threshold models of collective behavior. *American Journal of Sociology, 83*(6), 1420–1443.

Guo, L., Lin, J.-. H., Guo, Q., & Liu, J.-. G. (2016). Identifying multiple influential spreaders in term of the distance-based coloring. *Physics Letters A, 380*(7–8), 837–842.

Heidari, M., Asadpour, M., & Faili, H. (2015). SMG: Fast scalable greedy algorithm for influence maximization in social networks. *Physica A: Statistical Mechanics and its Applications, 420*, 124–133.

Huang, C.-. Y., Lee, C.-. L., Wen, T.-. H., & Sun, C.-. T. (2013). A computer virus spreading model based on resource limitations and interaction costs. *Journal of Systems and Software, 86*(3), 801–808.

Jalili, M., & Perc, M. (2017). Information cascades in complex networks. *Journal of Complex Networks, 5*(5), 665–693.

Jayakumar, N., Subramanian, S., Ganesan, S., & Elanchezhian, E. (2016). Gray wolf optimization for combined heat and power dispatch with cogeneration systems. *International Journal of Electrical Power and Energy Systems, 74*, 252–264.

Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical Review, 106*(4), 620.

Jiang, Q., Song, G., Gao, C., Wang, Y., Si, W., & Xie, K. (2011). Simulated annealing based influence maximization in social networks. In *Proceedings of the twenty–fifth AAAI conference on artificial intelligence*.

Kempe, D., Kleinberg, J., & Tardos, É. (2003). Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining*.

Kimura, M., Saito, K., Nakano, R., & Motoda, H. (2010). Extracting influential nodes on a social network for information diffusion. *Data Mining Knowledge Discovery, 20*(1), 70.

Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., & Stanley, H. E. (2010). Identification of influential spreaders in complex networks. *Nature Physics, 6*(11), 888.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., & Glance, N. (2007). Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining*.

Lu, F., Zhang, W., Shao, L., Jiang, X., Xu, P., & Jin, H. (2017). Scalable influence maximization under independent cascade model. *Journal of Network and Computer Applications, 86*, 15–23.

Meo, P. D., Musial-Gabrys, K., Rosaci, D., Sarne, G. M., & Aroyo, L. (2017). Using centrality measures to predict helpfulness-based reputation in trust networks. *ACM Transactions on Internet Technology, 17*(1), 8.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Gray wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Nowzari, C., Preciado, V. M., & Pappas, G. J. (2016). Analysis and control of epidemics: A survey of spreading processes on complex networks. *IEEE Control Systems Magazine, 36*(1), 26–46.

Peng, S., Yu, S., & Yang, A. (2014). Smartphone malware and its propagation modeling: A survey. *IEEE Communications Surveys and Tutorials, 16*(2), 925–941.

Pradhan, M., Roy, P. K., & Pal, T. (2016). Gray wolf optimization applied to economic load dispatch problems. *International Journal of Electrical Power and Energy Systems, 83*, 325–334.

Preciado, V. M., Zargham, M., Enyioha, C., Jadbabaie, A., & Pappas, G. (2013). Optimal vaccine allocation to control epidemic outbreaks in arbitrary networks. In *Proceedings of the 52nd IEEE conference on decision and control*.

Sabidussi, G. (1966). The centrality index of a graph. *Psychometrika, 31*(4), 581–603.

Sheikhahmadi, A., Nematbakhsh, M. A., & Shokrollahi, A. (2015). Improving detection of influential nodes in complex networks. *Physica A: Statistical Mechanics & its Applications, 436*, 833–845.

Sheikhahmadi, A., Nematbakhsh, M. A., & Zareie, A. (2017). Identification of influential users by neighbors in online social networks. *Physica A: Statistical Mechanics & its Applications, 486*, 517–534.

Sun, Y. (2016). Optimal selection of nodes to propagate influence on networks. *The European Physical Journal B, 89*(11), 253.

Wang, X., Su, Y., Zhao, C., & Yi, D. (2016). Effective identification of multiple influential spreaders by DegreePunishment. *Physica A: Statistical Mechanics and its Applications, 461*, 238–247.

Wang, Y., Cong, G., Song, G., & Xie, K. (2010). Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining*.

Yuan, X., Sun, Q., & Li, J. (2017). CNTE: A node centrality-based network trust evaluation method. In *Proceedings of the international conference on collaborative computing: networking, applications and worksharing*.

Zareie, A., & Sheikhahmadi, A. (2018). A hierarchical approach for influential node ranking in complex social networks. *Expert Systems with Applications, 93*, 200–211.

Zareie, A., Sheikhahmadi, A., & Jalili, M. (2019a). Identification of influential users in social networks based on users' interest. *Information Sciences, 493*, 217–231.

Zareie, A., Sheikhahmadi, A., & Jalili, M. (2019b). Influential node ranking in social networks based on neighborhood diversity. *Future Generation Computer Systems, 94*, 120–129.

Zareie, A., Sheikhahmadi, A., & Khamforoosh, K. (2018). Influence maximization in social networks based on topsis. *Expert Systems with Applications, 108*, 96–107.