Journal of Ambient Intelligence and Humanized Computing (2023) 14:3767–3779
https://doi.org/10.1007/s12652-021-03510-4

ORIGINAL RESEARCH

A quick GRASP-based method for influence maximization in social networks

Isaac Lozano-Osorio[1] · Jesús Sánchez-Oro[1] · Abraham Duarte[1] · Óscar Cordón[2]

Received: 4 February 2021 / Accepted: 13 September 2021 / Published online: 30 September 2021

Abstract
The evolution and spread of social networks have attracted the interest of the scientific community in the last few years. Specifically, several new interesting problems, which are hard to solve, have arisen in the context of viral marketing, disease analysis, and influence analysis, among others. Companies and researchers try to find the elements that maximize profit, stop pandemics, etc. This family of problems is collected under the term Social Network Influence Maximization problem (SNIMP), whose goal is to find the most influential users (commonly known as seeds) in a social network, simulating an influence diffusion model. SNIMP is known to be an $\mathcal{NP}$-hard problem and, therefore, an exact algorithm is not suitable for solving it optimally in reasonable computing time. The main drawback of this optimization problem lies on the computational effort required to evaluate a solution. Since each node is infected with a certain probability, the objective function value must be calculated through a Monte Carlo simulation, resulting in a computationally complex process. The current proposal tries to overcome this limitation by considering a metaheuristic algorithm based on the Greedy Randomized Adaptive Search Procedure (GRASP) framework to design a quick solution procedure for the SNIMP. Our method consists of two distinct stages: construction and local search. The former is based on static features of the network, which notably increases its efficiency since it does not require to perform any simulation during construction. The latter involves a local search based on an intelligent neighborhood exploration strategy to find the most influential users based on swap moves, also aiming for an efficient processing. Experiments performed on 7 well-known social network datasets with 5 different seed set sizes confirm that the proposed algorithm is able to provide competitive results in terms of quality and computing time when comparing it with the best algorithms found in the state of the art.

Keywords Information systems · Social networks · Influence maximization · Network science · Viral marketing · GRASP

1 Introduction

Nowadays, millions of users are involved in social networks (SNs), growing exponentially the number of active users. This growth is extended to the amount of behavioral data and, therefore, all classical network-related problems are becoming computationally harder. SNs can be defined as the representation of social interactions that can be used to study the propagation of ideas, social bond dynamics, disease propagation, viral marketing, or advertisement, among others (D'angelo et al. 2009; Klovdahl 1985; Barabási and Pósfai 2016; Reza et al. 2014).

SNs are used not only for spreading positive information but also malicious information. In general, research devoted to maximize the influence of positive ideas is called Influence Maximization (Nguyen Hung et al. 2016). Thus, solving successfully this problem allows the decision-maker to decide the best way to propagate information about products and/or services. On the contrary, SNs can be also used for the diffusion of malicious information like derogatory rumors, disinformation, hate speech, or fake news. These examples motivate research about how to reduce the

✉ Óscar Cordón
ocordon@decsai.ugr.es

Isaac Lozano-Osorio
isaac.lozano@urjc.es

Jesús Sánchez-Oro
jesus.sanchezoro@urjc.es

Abraham Duarte
abraham.duarte@urjc.es

1 Universidad Rey Juan Carlos, Móstoles, Spain

2 Universidad de Granada, Granada, Spain

influence of negative information. This family of problems is usually known as Influence Minimization (Khalil Elias et al. 2013; Luo et al. 2014; Xinjue et al. 2018; Qipeng et al. 2015).

A SN is usually modeled with a graph $G(V, E)$ where the set of nodes V represents the users and each relation between two users is modeled as a pair $(u, v) \in E$, with $u, v \in V$ indicating that user u is connected to or even can transmit information to user v. Kempe et al. (2003) originally formalized the influence model to analyze how the information is transmitted among the users of a SN. Given a SN with $|V| = n$ nodes where the edges (relational links) represent the spreading or propagation process on that network, the task is to choose a seed node set S of size $k < n$ with the aim of maximizing the number of nodes in the network that are influenced by the seed set S. This results in a combinatorial optimization problem known as the Social Network Influence Maximization problem (SNIMP).

The evaluation of the influence of a given seed set S requires the definition of an Influence Diffusion Model (IDM) (Kempe et al. 2015). This model is responsible for deciding which nodes are affected by the information received from their neighboring nodes in the SN. The most extended IDMs are: Independent Cascade Model (ICM), Weighted Cascade Model (WCM), and Linear Threshold Model (LTM). All of them are based on assigning an influence probability to each relational link in the SN. ICM, which is one of the most used IDMs, considers that the influence probability is the same for each link, and it is usually a small probability, being 1% a widely accepted value. On the contrary, WCM considers that the probability of a user v for being influenced by user u is proportional to the in-degree of user v, i.e., the number of users that can eventually influence user v. Therefore, the probability of influencing user v is defined as $1/d_{in}(v)$, where $d_{in}(v)$ is the in-degree of user v. The latter model, LTM, requires a specific activation weight for each link in the SN. Given those weights, a user will be influenced if and only if the sum of the weights of its neighbors if larger than or equal to a given threshold. In this paper we consider the ICM since it is one of the most popular IDMs in the literature. In particular, ICM views influence as being transmitted through the network in a tree-like fashion, where the seed nodes are the roots.

The SNIMP then involves finding a seed set S, with $|S| = k$ (where k is an input parameter), that maximizes the number of users influenced and, as a consequence, the spread of information through the network. In mathematical terms,

$$S^\star \leftarrow \arg\max_{S \in \mathcal{S}} ICM(G, S, p, ev)$$

where $\mathcal{S}$ is the set of all possible solutions (i.e., seed set setups), p is the probability of a user to be influenced, and ev is the number of iterations of the Monte Carlo simulation used to run the ICM (see Sect. 3.1).

The SNIMP was initially formulated in Richardson et al. (2003) and it was later proven to be $\mathcal{NP}$-hard for most IDMs in Kempe et al. (2015). As with many other $\mathcal{NP}$-hard problems, heuristic and metaheuristic algorithms, such as greedy and evolutionary algorithms, have been considered to solve the problem by effectively exploring the solution space, avoiding the analysis of every possible subset of nodes (Banerjee et al. 2020).

This work presents a novel metaheuristic approach for dealing with the SNIMP, allowing us to find high quality solutions in short computing time. Our main goal is to design an efficient algorithm where the use of Monte Carlo simulation required for the IDM application is minimized, thus increasing the efficiency of the algorithm. To do so, we make use of the Greedy Randomized Adaptive Search Procedure (GRASP) framework, characterized for its efficiency when designing solutions for $\mathcal{NP}$-hard combinatorial optimization problems. Our procedure is based on two stages. On the one hand, a greedy constructive procedure based on the 2-step neighborhood which is randomized to diversify the search with the aim of exploring a wider portion of the solution space. On the other hand, we introduce an efficient local search method. Specifically, it relies on an intelligent neighborhood exploration strategy for finding local optima with respect to the constructed solutions. The proposed procedure is validated over a set of 35 instances widely used in the context of social influence maximization, and benchmarked against both the classical methods based on greedy hill-climbing strategies (Goyal et al. 2011; Leskovec et al. 2007) and the state-of-the-art solution procedure for SNIMP based on particle swarm optimization (Banerjee et al. 2020). The results obtained clearly demonstrate the efficacy of the proposed methodology.

The remainder of the work is structured as follows. Section 2 reviews the related literature, detailing the different approaches followed to deal with different problems derived from the social influence maximization. The proposed approach is described in Sect. 3, where Sect. 3.1 introduces the influence diffusion model selected in this research, Sect. 3.2 presents the construction method proposed for providing high quality initial solutions, and Sect. 3.3 describes the local search proposed for finding local optimum with respect to a given neighborhood structure. Section 4 presents the experimental results considering a public dataset which has been previously used for this task, divided into preliminary experiments devoted to adjust the search parameters (Sect. 4.1), and final experiments to perform a competitive testing to evaluate the quality of the proposal (Sect. 4.2). Finally, Sect. 5 draws some conclusions derived from this research.

2 Literature review

In this section, we introduce some related work about SNIMP and IDM as well as a brief survey of existing methods for solving this problem, either based on heuristics or computational intelligence algorithms.

Richardson et al. (2003) initially formulated the problem of selecting target nodes in SNs. However, Kempe et al. (2003) were the first to solve the SNIMP formulating it as a discrete optimization problem. It has been shown that the SNIMP is $\mathcal{NP}$-hard (Kempe et al. 2015). Kempe et al. (2003) proposed a greedy hill-climbing algorithm with an approximation of $1 - 1/e - \epsilon$, being e the base of the natural logarithm and ϵ any positive real number. This result indicates that the algorithm is able to find solutions which are always within a factor of at least 63% of the optimal value under the three IDMs described in Sect. 1.

As a consequence of the computational effort required to evaluate the ICM, Kempe et al. (2003) also proposed several greedy heuristics based on SN analysis metrics such as degree and closeness centrality (Stanley and Katherine 1994). These methods only require one run of a Monte Carlo simulation to validate the single solution obtained using heuristic functions, thus increasing the efficiency at the cost of a loss of efficacy. When the considered metric is the degree of the node, the algorithm is called *high-degree heuristic*.

Several extensions of those first greedy algorithms were later proposed. In particular, Leskovec et al. (2007) introduced the Cost-Effective Lazy Forward (*CELF*) selection which exploited the submodularity property to significantly reduce the run time of the greedy hill-climbing algorithm, becoming over 700 times faster than the original procedure. The rationale is that the expansion of each node is computed a priori and it only needs to be recomputed for a few nodes. Meanwhile, Chen et al. (2009) used the concept of degree-discount heuristics to optimize the high-degree heuristic. The greedy selection function considers the redundancy between likely influenced nodes and does not include those reached by the already selected seed nodes to provide a better estimation of the total spread.

In Goyal et al. (2011) a new algorithm called *CELF++* was proposed with the aim of improving the efficiency of the original CELF. It leans on the property of submodularity of the spread function for IDM, avoiding unnecessary computations. According to the authors it is 35-55% faster than *CELF*.

A large number of works have been developed in the area since those first proposals (Şimşek and Kara 2018). Different kinds of heuristic and metaheuristic algorithms have been considered to solve the SNIMP. Table 1 summarizes the most recent approaches, including the algorithm type and the specific IDM considered.

Analyzing previous studies we can conclude that more complex metaheuristic approaches usually result in better solutions than simple greedy approaches. Yang and Weng (2012) proposed an Ant Colony Optimization (ACO) algorithm based on a parameterized probabilistic model to address the SNIMP. They used the degree centrality, distance centrality, and simulated influence methods for determining the heuristic values.

Meanwhile, the method based on Simulated Annealing (SA) presented in Li et al. (2017) applied two heuristic methods to accelerate the convergence process of SA, along with a new method of computing influence spread to speed up the proposed algorithm. In Bucur et al. (2017), an Evolutionary Multi-objective Optimization (EMO) algorithm (Lamont et al. 2007) was proposed for SNIMP, where the two considered objectives were maximizing the influence of a seed set and minimizing the number of nodes in the seed set jointly.

As said before, some heuristics have been proposed as time-saving solutions for greedy decisions: random, degree, and centrality (Kempe et al. 2003). The random heuristic selects nodes randomly, without considering node influence, to form the seed set in the network. Degree and centrality heuristics derive from the definition of the node influence in SN analysis (Stanley and Katherine 1994). Degree centrality heuristic usually produces less accurate results to the SNIMP. Alternatively, the high-degree heuristic targets the SNIMP by taking into account prior knowledge of the node's neighbors (Chen et al. 2009).

Recently, a complete survey on SNIMP has been presented in Banerjee et al. (2020). In that work, authors experimentally compare the results obtained by the most recent algorithms. The survey concludes that the particle swarm optimization approach by Gong et al. (2016) obtains the best results in the literature, so we will use that algorithm to benchmark our proposal. This survey has become one of the most relevant research in the area of influence maximization problems.

The aforementioned metaheuristic methods are able to obtain good results but usually require large computational times. Our proposal considers a method combining the use of heuristic functions and Monte Carlo simulations for limiting the number of ICM evaluations that tries to balance the quality of the obtained solutions and the required computing time. With the aim of reducing the required computing time, we consider a GRASP, combining good heuristic solutions that are quickly generated and an efficient local search method which minimizes the number of Monte Carlo evaluations to further improve the initial solution. The use of GRASP in Graph Theory and Network Science has led to several successful research in the last years (Pérez-Peló et al. 2019, 2020; Gil-Borrás et al. 2020).

Table 1 Summary of the literature published for the SNIMP in the last 20 years

References	Algorithm	IDM
Kempe et al. (2003)	Greedy	ICM-LTM
Chen et al. (2009)	Greedy	ICM
Lappas et al. (2010)	Dynamic Programming	ICM
Goyal et al. (2011)	Greedy	ICM
Yang and Weng (2012)	Ant Colony	ICM
Nguyen and Zheng (2013)	Greedy	ICM
Jiaguo et al. (2014)	Greedy	ICM
Li et al. (2014)	Greedy	Polarity-Related-ICM
Liu et al. (2014)	Greedy	Latency Aware-ICM
Lee and Chung (2015)	Greedy	ICM
Song et al. (2015)	Greedy	ICM
Bucur and Iacca (2016)	Genetic	ICM
Gong et al. (2016)	Particle Swarm	ICM
Ok et al. (2016)	Greedy	Independent Poisson Clock
Tong et al. (2016)	Greedy	ICM
Zhang et al. (2016)	Greedy	LTM-ICM
Bucur et al. (2017)	Multi-objective evolutionary	ICM
Li et al. (2017)	Simulated Annealing	Polarity-Related-ICM
Peng et al. (2017)	Greedy	SI
Tong et al. (2017)	Greedy	ICM
Zhang et al. (2017)	Genetic	LTM
Bucur et al. (2018a)	Multi-objective evolutionary	ICM
Bucur et al. (2018)	Multi-objective evolutionary	ICM
Samadi et al. (2018)	Mixed Integer Programming	Partial Parallel Cascade
Liu et al. (2019)	Evolutionary	SIR epidemic spreading model
Salavati and Abdollahpouri (2019)	Ant Colony	ICM
Robles et al. (2020)	Multi-objective evolutionary	Viral marketing model

3 Algorithmic approach

The method presented in this work aims at finding high quality solutions in reasonable computing time. In order to do so, we propose a metaheuristic algorithm. Metaheuristics are one of the most extended techniques for solving hard optimization problems. They are able to guide the search in order to escape from local optima, with the goal of finding high quality solutions in a reduced computing time.

GRASP is a metaheuristic framework developed in the late 1980s Feo and Resende Mauricio (1989) and formally introduced in Feo and Resende Mauricio (1994). We refer the reader to Resende Mauricio et al. (2010), Resende Mauricio and Celso (2013) for a complete survey of the last advances in this methodology. GRASP is a multi-start framework divided into two distinct stages. The first one is a greedy, randomized, and adaptive construction of a solution. The second stage applies an improvement method to obtain a local optimum with respect to a certain neighborhood, starting from the constructed solution. This methodology is able to find a trade-off between the diversification of the randomized construction phase and the intensification of the local search procedure, allowing the algorithm to escape from local optima and perform a wider exploration of the search space. These two phases are repeated until a termination criterion is met, returning the best solution found during the search.

3.1 Influence diffusion model

Before defining the algorithmic proposal, it is necessary to provide a formal definition of the IDM considered in this work, which is the ICM introduced in Sect. 1. Due to the probabilistic nature of ICM, the most extended way of evaluating the spread is by conducting a Monte Carlo simulation. However, even a single iteration of the simulation in ICM is rather time-consuming. Algorithm 1 shows the pseudocode of the Monte Carlo simulation to evaluate the spread of information through a SN named G given a seed set S. Specifically, it receives four input parameters: the graph which models the SN, a tentative solution, the probability of a user to be influenced, and the number of iterations of the corresponding Monte Carlo simulation.

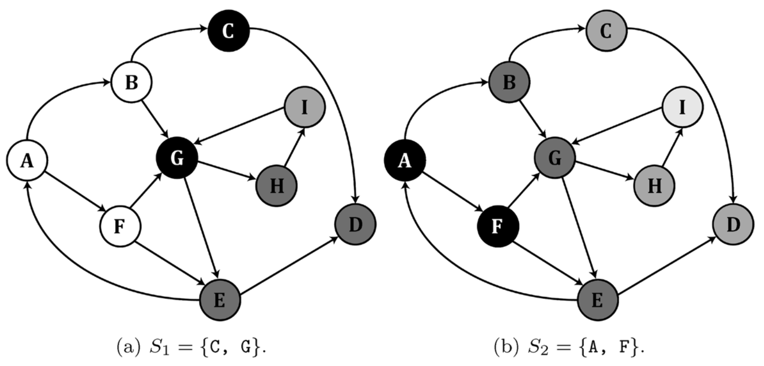

Fig. 1 SN with 9 nodes and 13 edges. Two feasible solutions S_1 and S_2 are represented, each of them resulting in a different set of influenced users

Algorithm 1 $ICM(G = (V, E), S, p, ev)$

```
1: $I \leftarrow \emptyset$
2: for $i \in 1 \ldots ev$ do
3:     $A^\star \leftarrow S$
4:     $A \leftarrow S$
5:     while $A \neq \emptyset$ do
6:         $B \leftarrow \emptyset$
7:         for $v \in A$ do
8:             for $(u, v) \in E$ do
9:                 if $rnd(0, 1) \leq p$ then
10:                    $B \leftarrow B \cup \{u\}$
11:                end if
12:            end for
13:        end for
14:        $A^\star \leftarrow A^\star \cup B$
15:        $A \leftarrow B$
16:    end while
17:    $I \leftarrow I + |A^\star|$
18: end for
19: return $I/ev$
```

The algorithm starts by initializing the set which stores the number of infected users (step 1). It then performs a number of predefined iterations *ev* (steps $2 - 18$), finding in each iteration which are the influenced nodes by the given seed set *S*. Initially, the set of nodes $A^\star$ reached by the initial seed set, *S*, is actually the seed set (step 3). Then, the method iterates until no new nodes are influenced (steps 5–16). In each iteration of the inner for-loop, the neighbors of each node reached in the previous one are traversed (steps 8–12). For each neighbor, a random number is generated. If this number is smaller than the probability of infection *p*, then the neighboring node becomes infected (steps 9–11). At the end, the set of infected nodes is updated (step 14) as well as the nodes infected in the previous iteration (step 15). Finally, the algorithm returns the mean number of infected nodes among all the simulations performed, i.e., *I* divided by *ev* (step 19). Notice that this value is considered as the objective function to be optimized when solving the SNIMP. That is, the seed set maximizing the *spread value* over the network would compose the optimal solution to the problem. It is worth mentioning that, as infection is a stochastic process, the ICM must be run several times (*ev* in our case) to achieve an appropriate estimation, thus resulting in a Monte Carlo simulation.

In order to illustrate the evaluation of a solution under the ICM, Fig. 1 shows an example of a SN with 9 nodes and 13 directed edges, where each pair (u, v) denotes that the user v may be influenced by u. Information represents anything that can be passed across connected peers within a network. The influence level given by a node is determined by the adoption or propagation process. Let us consider $k = 2$ for this example graph.

Figure 1a shows solution S_1 where the seed set is conformed with nodes C and G. Without loss of generality we will assume, for this example, that $p = 1$ (i.e., all the nodes are always infected by their neighbors). Simulating the diffusion model we can see how nodes D, E, and H are directly influenced by the seed set. After that, node H influences node I. Different levels of influence are represented by a gray gradient from black to white. Therefore, if we consider a single evaluation of the Monte Carlo simulation, the objective function value of S_1 is $ICM(G, S_1, p, 1) = 6$. Figure 1b depicts solution $S_2 = \{\mathtt{A}, \mathtt{F}\}$. Similarly to Fig. 1a, a gray gradient indicates the process of influence over the network, resulting in an objective function value of $ICM(G, S_2, p, 1) = 9$, since all nodes are influenced. Notice that, following this evaluation, S_2 is better than S_1. However, it is important to remark that a single iteration for the Monte Carlo simulation is not significant, so we have decided to perform ev =100 evaluations of the simulation as it is customary in the literature for the SNIMP (Bucur and Iacca 2016; Bucur et al. 2018). Additionally, we set $p = 0.01$ as stated in previous works Bucur and Iacca 2016; Kempe et al. 2003; Gong et al. 2016. Then, for the sake of simplicity, we refer to $ICM(G, S, 0.01, 100)$ as $ICM(G, S)$ in the remaining of the paper.

(*RCL*) with the most promising nodes (step 8). This threshold directly depends on the value of the input parameter α, which is in the range [0, 1]. Notice that this parameter indicates the greediness or randomness of the constructive procedure. On the one hand, if $\alpha = 0$, then the threshold is evaluated as $g_{\max}$, becoming a totally greedy algorithm (i.e., the *RCL* only includes the best choice in each iteration). On the other hand, if $\alpha = 1$ then $\mu = g_{\min}$, resulting in a completely random method (i.e., the *RCL* includes every feasible choice in each iteration). Since this parameter is tuned experimentally, we refer the reader to Sect. 4 to analyze the experiments performed to select the best value for α. Finally, the next node is selected at random from the *RCL* (step 9), including it in the solution (step 10) and updating the *CL* (step 11). The method ends when k elements are included in the seed set, returning the constructed solution S (step 13).

Algorithm 2 $Construct(G = (V, E), \alpha)$

```
1:  $v \leftarrow rnd(V)$
2:  $CL \leftarrow V \setminus \{v\}$
3:  $S \leftarrow \{v\}$
4:  while $|S| < k$ do
5:      $g_{\min} \leftarrow \min_{u \in CL} g(u)$
6:      $g_{\max} \leftarrow \max_{u \in CL} g(u)$
7:      $\mu \leftarrow g_{\max} - \alpha \cdot (g_{\max} - g_{\min})$
8:      $RCL \leftarrow v \in CL : g(v) \geq \mu$
9:      $u \leftarrow rnd(RCL)$
10:     $S \leftarrow S \cup \{u\}$
11:     $CL \leftarrow CL \setminus \{u\}$
12: end while
13: return $S$
```

3.2 Construction phase

The construction phase of GRASP is designed to generate an initial solution and it is usually guided by a greedy selection function which helps the constructive method to select the next elements to be included in the partial solution (see Algorithm 2).

In order to favor diversification, the first node to be included in the solution S is selected at random from the set of SN nodes V (step 1). The candidate list *CL* is created with all the nodes but v (step 2) and the node v is included as the only node in the initial solution S (step 3). Then, the constructive method iteratively adds new elements to the solution until it becomes feasible, being composed of k nodes (steps 4–12). In each iteration, the minimum and maximum value of the greedy heuristic function is evaluated (steps 5–6). Then, a threshold μ is calculated (step 7), which is required for creating the Restricted Candidate List

The greedy heuristic function g used in steps 5-6 is one of the key features when designing a constructive procedure in the context of GRASP. In this work we propose two different greedy functions to generate initial solutions. The first one, named g_{ne}, is a heuristic based on the first and second degree neighbors of a given node (usually known as 2-step in SN analysis Stanley and Katherine 1994). Given a node u, its out-degree is defined as $d_u^+ = |N_u^+ = |$, where $N_u^+ = \{w \in V : (u, w) \in E\}$ is the set of adjacent nodes to u.

Thus, the first greedy function calculates the sum of the out-degree of u plus the out-degree of its neighbors. This heuristic function is based on the two-hop area (Sen et al. 2014) and three-degree theory (Christakis and Fowler 2009) hypothesis, indicating that there exists an intrinsic decay when increasing the maximum neighborhood level. More formally,

$$g_{ne}(u) = d_u^+ + \sum_{v \in N_u^+} d_v^+$$

If we analyze this metric over the graph depicted in Fig. 1, the value of g_{ne} over vertex $\mathtt{F}$, for instance, is evaluated as $g_{ne}(\mathtt{F}) = d^+_{\mathtt{F}} + d^+_{\mathtt{E}} + d^+_{\mathtt{G}} = 2 + 2 + 2 = 6$. In order to reduce the computational effort of updating the value of this greedy function for each node, the method updates the value just for the affected nodes, which are those nodes to which the selected one is directly connected. For instance, in case of selecting node $\mathtt{F}$, only nodes $\mathtt{E}$ and $\mathtt{G}$ must be updated, by subtracting the out-degree of the selected node $\mathtt{F}$. This efficient update mechanism allows the algorithm to minimize the relevance of those vertices that are directly influenced by one of the selected nodes.

The second greedy function, named g_{cc}, considers the nodes clustering coefficient as a heuristic value. It estimates the likelihood that nodes in a graph tend to cluster together. In mathematical terms, the clustering coefficient is defined as (in a directed network):

$$g_{cc}(u) = \frac{\left|\{(v,w) \in E \,:\, v,w \in N^+_u\}\right|}{d^+_u \cdot (d^+_u - 1)}$$

It is worth mentioning that, to speed up this computation, every value is pre-calculated before the execution of the algorithm. We refer the reader to Watts and Strogatz (1998) for a detailed description of the implementation of this metric.

Let us illustrate how we can evaluate this value with an example. Considering again the SN depicted in Fig. 1, the evaluation of the clustering coefficient of node $\mathtt{F}$ is performed as $g_{cc} = \frac{|(\mathtt{G},\mathtt{E})|}{d^+_{\mathtt{F}} \cdot (d^+_{\mathtt{F}} - 1)} = \frac{1}{2} = 0.5$.

Given g_{ne} and g_{cc} as greedy functions, we propose two different constructive procedures C_{ne} and C_{cc}, each one based on a different greedy function. The impact and influence of each constructive procedure in the generated solutions will be deeply analyzed in Sect. 4.

3.3 Local search phase

The second phase of GRASP involves improving the solution generated by the constructive procedure in each iteration with the aim of reaching a local (ideally global) optimum. In the context of GRASP, this phase can be accomplished by using simple local search procedures or more complex heuristics like Tabu Search or even a hybrid metaheuristic (Martí et al. 2018). The high complexity of the problem under consideration has led us to propose a low time consuming local search procedure.

Before defining a local search method, it is necessary to introduce the neighborhood to be explored. The neighborhood of a solution S is defined as the set of solutions that can be reached by performing a single move over S. In the context of SNIMP, we propose a swap move $Swap(S, u, v)$ where node u is removed from the seed set, being replaced by v, with $u \in S$ and $v \notin S$. This swap move is formally defined as:

Table 2 Nodes and edges of the instances used in this work

Instance	Nodes	Edges	References
WikiVote	7115	103689	Bucur and Iacca (2016), Lawyer (2015)
ca-AstroPh	18772	198110	Liu et al. (2019), Lawyer (2015)
ca-CondMat	23133	93497	Liu et al. (2019), Lawyer (2015)
cit-HepPh	34546	421578	Liu et al. (2019), Lawyer (2015)
email-Enron	36692	183831	Liu et al. (2019)
p2p-Gnutella31	62586	147892	Liu et al. (2019), Lawyer (2015)
email-EuAll	265214	420045	Liu et al. (2019), Lawyer (2015)

$$Swap(S, u, v) = S \setminus \{u\} \cup \{v\}$$

Thus, the neighborhood N_s of a given solution S consist of the set of solutions that can be reached from S by performing a single swap move. More formally,

$$N_s(S) = \{Swap(S, u, v) \quad \forall\, u \in S \wedge \forall\, v \in V \setminus S\}$$

The next step to define the proposed local search procedure consists in indicating the way in which the neighborhood is explored. Even considering an efficient implementation of the objective function evaluation, the vast size of the resulting neighborhood, $k \cdot (n - k)$, makes the complete exploration of the neighborhood not suitable for the SNIMP. Therefore, an intelligent neighborhood exploration strategy is presented with the aim of reducing the number of solutions explored within each neighborhood. This reduction in the size of the search space is performed by exploring just a small fraction, δ, of the available nodes for the swap node.

Since we are limiting the number of nodes considered in the local search approach, it is recommended to select the most promising ones to be involved in the swap moves. In the context of SNIMP, a node with a large out-degree can eventually influence a larger amount of nodes. Following this idea, we sort the candidate nodes to be included in the seed set in descending order with respect to their out-degree, while the candidate nodes to be removed from the seed set are sorted in ascending order with respect to their out-degree. Sect. 4 details the impact of the number of selected nodes, determined by δ, in the results obtained with the local search procedure.

Finally, we need to indicate which moves will be accepted during the search. In particular, two strategies are usually considered: Best Improvement and First Improvement

Table 3 Results of the constructive procedure when generating 100 solutions, considering different α values for both heuristic functions

Heuristic	α	Avg.	Time (s)	Dev (%)	#Best
C_{ne}	0.25	236.62	**6.23**	15.27	0
	0.50	258.57	7.07	2.17	1
	0.75	262.11	7.15	0.31	6
	RND	**262.38**	6.55	**0.11**	**8**
C_{cc}	0.25	96.49	77.21	63.24	0
	0.50	83.06	77.21	65.41	0
	0.75	93.18	78.96	64.57	0
	RND	95.04	77.21	63.24	0

Best results are highlighted with bold font

(Hansen and Mladenović 2006). On the one hand, the former explores, in each iteration, the complete neighborhood, moving to the best solution in it. On the other hand, the latter moves to the first solution that achieves an improvement in the objective function value, without requiring to explore the complete neighborhood. Due to the computational effort required to evaluate a solution for the SNIMP, we propose a First Improvement approach, which does not need to explore all the solutions in the neighborhood, thus reducing the number of objective function evaluations required and consequently the overall run time.

Notice that the objective function evaluation consists in a Monte Carlo simulation, being the most computationally demanding part of the proposed algorithm. For this reason, the proposed local search aims to limit the number of required simulations, thus leading to a more efficient procedure.

4 Computational experiments and analysis of results

The aim of this section is to describe the computational experiments designed to evaluate the performance of the proposed algorithms and to analyze the obtained results. All the experiments have been performed in an Intel Core i7 (2.6 GHz) with 8GB RAM and the algorithms were implemented using Java 9. The source code has also been made publicly available.[1]

The set of SNs considered in this paper have been entirely obtained from the most relevant works found in the literature, in order to provide a fair comparison among the analyzed algorithms. All of them are publicly available in Stanford Network Analysis Project (SNAP): https://snap.stanford.edu/. Relevant information about these instances

[1] https://grafo.etsii.urjc.es/SNIMP.

Table 4 Influence of the number of nodes δ explored in each iteration of the local search procedure when coupled with the constructive procedure

δ	Avg.	Time (s)	Dev. (%)	#Best
10	272.28	47.28	0.89	5
20	272.46	**43.20**	0.56	**8**
30	**274.42**	101.31	**0.51**	4
40	273.72	69.17	0.54	4

Best results are highlighted with bold font

is collected in Table 2, where some papers in which each instance has been used are included.

First of all, it is important to indicate which values are used for the ICM algorithm with the corresponding Monte Carlo simulation. In all the experiments, as stated in Sect. 3.2, 100 simulations of the ICM are performed with a probability of influence of 0.01. These parameter values are the most extended ones in the related literature. The number of seed nodes k to conform a solution is selected in the range $k = \{10, 20, 30, 40, 50\}$ as stated in Bucur and Iacca (2016), Kempe et al. (2003), Salavati and Abdollahpouri (2019), thus obtaining $7 \cdot 5 = 35$ different problem instances (resulting from the combination of 7 networks and 5 seed set sizes).

The experiments are divided into two parts: preliminary and final experimentation. The former (Sect. 4.1) refers to those experiments performed to select the best parameters to set up our algorithm, while the latter (Sect. 4.2) validates the best configuration, comparing it with the best methods found in the state of the art.

All the experiments developed report the following metrics: Avg., the average objective function value (i.e., the number of influenced nodes, in average, after 100 simulations); Time (s), the average computing time required by the algorithm in seconds; Dev(%), the average deviation with respect to the best solution overall found in the experiment; and #Best, the number of times that the algorithm matches the best solution.

4.1 Preliminary experimentation to setup the final GRASP method

The preliminary experimentation has been performed with a small set of 10 instances out of 35 to avoid overfitting. This selection comprises, approximately 30% of the global set and it provides enough variability in instances and values of k.

The first preliminary experiment is designed to find the best value for the α parameter in each of the proposed constructive procedures (see Sect. 3.2). Table 3 shows the detailed results for each constructive procedure when considering $\alpha = \{0.25, 0.50, 0.75, RND\}$, where *RND* indicates

Table 5 Competitive testing of the proposed GRASP algorithm with respect to the best algorithms found in the literature: CELF, CELF++, and PSO

k		CELF		CELF++		PSO		GRASP	
	Name	Avg.	Time (s)	Avg.	Time (s)	Avg.	Time (s)	Avg.	Time (s)
10	CA-AstroPh	157.60	**2.51**	171.81	9.40	169.85	232.40	**187.47**	8.28
	CA-CondMat	35.73	**0.67**	35.73	2.15	33.40	4.60	**36.15**	2.56
	Cit-HepPh	46.63	**1.16**	46.63	3.29	35.27	1.71	**47.20**	4.20
	Email-Enron	383.95	**25.23**	469.63	87.68	465.24	1756.84	**489.67**	41.41
	Email-EuAll	132.96	**6.03**	130.28	307.98	107.41	37.42	**144.57**	24.42
	Wiki-Vote	108.50	**0.39**	108.50	1.00	92.16	16.40	**109.10**	6.32
	p2p-Gnutella31	16.24	**1.46**	16.23	7.83	13.38	0.95	**16.27**	1.63
20	CA-AstroPh	222.63	**2.69**	234.36	9.76	222.92	889.79	**259.25**	18.53
	CA-CondMat	59.72	**0.66**	59.87	2.13	45.46	8.67	**61.05**	6.00
	Cit-HepPh	81.75	**1.11**	81.75	3.25	68.51	2.58	**82.11**	18.97
	Email-Enron	451.24	**25.71**	547.96	88.47	544.57	4394.46	**589.65**	74.23
	Email-EuAll	214.66	**5.68**	214.54	303.01	162.32	99.98	**224.10**	28.88
	Wiki-Vote	162.49	**0.49**	162.49	1.45	141.66	41.44	**165.32**	26.03
	p2p-Gnutella31	30.82	**1.30**	30.86	7.43	24.69	0.99	**30.92**	3.80
30	CA-AstroPh	266.77	**2.85**	276.69	10.48	259.90	1005.17	**312.68**	51.32
	CA-CondMat	80.87	**0.70**	82.18	2.30	66.27	11.09	**82.54**	14.11
	Cit-HepPh	113.39	**1.16**	113.39	3.45	86.22	3.69	**113.63**	42.30
	Email-Enron	501.78	**25.49**	608.63	88.62	553.25	7594.67	**652.48**	140.71
	Email-EuAll	277.40	**5.86**	275.36	298.66	212.84	183.58	**281.30**	59.58
	Wiki-Vote	208.18	**0.64**	208.18	2.03	150.40	97.75	**214.97**	80.83
	p2p-Gnutella31	44.75	**1.24**	**44.81**	7.42	35.30	1.34	**44.81**	6.08
40	CA-AstroPh	319.52	**3.11**	302.86	11.58	288.92	1492.82	**360.34**	66.97
	CA-CondMat	100.96	**0.76**	101.80	2.54	75.61	17.40	**104.38**	16.37
	Cit-HepPh	140.63	**1.27**	140.63	3.81	113.46	4.94	**141.20**	58.03
	Email-Enron	549.64	**25.95**	658.38	92.09	634.58	9032.87	**705.03**	216.65
	Email-EuAll	323.85	**6.17**	312.47	302.47	258.46	230.12	**337.39**	165.23
	Wiki-Vote	246.02	**0.83**	246.02	2.83	182.88	115.05	**252.15**	34.60
	p2p-Gnutella31	58.26	**1.28**	58.22	7.48	51.26	1.85	**58.37**	12.69
50	CA-AstroPh	361.51	**3.50**	338.28	13.11	340.54	2267.98	**399.92**	132.35
	CA-CondMat	119.29	**0.86**	120.72	2.87	106.10	10.85	**124.57**	26.51
	Cit-HepPh	165.47	**1.38**	165.47	4.26	126.77	6.35	**166.77**	65.20
	Email-Enron	597.26	**27.02**	680.29	96.71	662.67	10063.47	**744.38**	157.26
	Email-EuAll	361.51	**6.68**	357.43	304.03	258.15	321.81	**375.03**	161.59
	Wiki-Vote	277.65	**1.09**	277.65	3.76	188.82	181.07	**287.66**	86.39
	p2p-Gnutella31	71.80	**1.34**	71.90	7.76	64.63	2.74	**72.08**	17.42
	G.Avg.	208.33	**5.55**	221.49	60.09	195.54	1146.71	**236.37**	39.04

Best results are highlighted with bold font

that the value of α is selected at random in each construction (thus allowing the algorithm for a higher and balanced search space diversification). The method is executed 100 independent times per instance, returning the best constructed solution.

As it can be drawn from the table, the best results are consistently provided by the greedy function based on the two-step neighbors, C_{ne}, both in quality and run time. In particular, the best results are obtained when considering $\alpha = RND$, with 8 best solutions and 0.11% of average deviation. The small deviation value indicates that, even in the cases in which it is not able to reach the best solution, it remains really close to it. Besides, it is the second best choice in terms of run time, with a slight difference with respect to the best option. Therefore, we select C_{ne} as the best constructive procedure with $\alpha = RND$.

The second preliminary experiment is devoted to analyze the influence of the number of explored nodes in the neighborhood for the local search phase. In particular, we have tested $\delta = \{10, 20, 30, 40\}$, being δ the number of nodes explored in each local search iteration. Table 4 shows the influence that the number of nodes explored δ has in the performance of the local search procedure. Notice that an independent local search is applied to each one of the 100 constructed solutions, returning the best solution found overall.

The obtained results show that the best δ value is 30, reaching the smallest deviation and the largest number of best solutions found, although in this case the results are more similar in every metric. However, it is more computationally demanding than the remaining values, being two or even three times slower than the other δ values. Furthermore, $\delta = 20$ is the quickest variant (almost 2.5 times faster than $\delta = 30$) and it presents a very good performance: a promising average objective function value and average deviation with respect to the best value (272.46 versus 274.42 and 0.56% versus 0.51%, respectively). For this reason, we select $\delta = 20$ as our design choice for the final algorithm.

It is important to remark the relevance of the introduced intelligent neighborhood exploration strategy in the performance of the whole algorithm. Specifically, the best identified local search method (with $\delta = 20$) spends 26.03 seconds on solving the smallest instance (WikiVote), obtaining an objective function value of 165.32. If we do not consider the δ parameter and execute an exhaustive local search, it needs 15793.09 seconds (i.e., more than 600 times longer) to find the local optima with a value of 165.59. We do not extend this experiment for the remaining instances since the computing time is unacceptable.

Having performed the preliminary experiments, we can conclude that the best results are obtained with the configuration greedy function=g_{ne}, $\alpha = RND$, and $\delta = 20$. These parameter values will be used to set up the final version of the algorithm.

4.2 Final experimentation to benchmark the final GRASP method with state-of-the-art results

In order to analyze the quality of the proposed algorithm, we perform a competitive testing with the best methods found in the state of the art by considering the complete set of 35 instances. In this experiment, three additional algorithms are considered: CELF (Leskovec et al. 2007), the well-known greedy hill-climbing algorithm; CELF++ (Goyal et al. 2011), the improved version of CELF (Leskovec et al. 2007); and PSO (Gong et al. 2016), the particle swarm optimization algorithm which is considered the state of the art for social influence analysis according to the recent experimental study developed in Banerjee et al. (2020). Table 5 collects the final results obtained in this competitive testing, where we report for each method and instance the value of the objective function (Avg.) and the associated computing time in seconds (Time(s)). Notice that Avg. is not an integer value since it is the average value of the 100 runs of the ICM in the Montecarlo simulation. We have also included a final row in this table (G.Avg.) with the average values of the objective function and Time (s), computed across the set of 35 instances.

We would like to first highlight the results obtained with PSO, since it is ranking the last one even being considered the state of the art for this problem. The rationale behind this is that the original work (Gong et al. 2016) considers small size instances (from 410 to 15233 nodes) and the quality of the solutions provided by PSO deteriorates when the instance size increases, as it can be derived from Table 5. Meanwhile, CELF and its improved version CELF++ are able to reach better solutions, still being competitive with the state-of-the-art algorithms. However, only CELF++ is able to match the best-known solution in 1 instance (out of 35). Finally, the best results are obtained with the proposed GRASP algorithm, which is able to reach the best solution found in all the 35 instances. Furthermore, the computing time is smaller than the second best algorithm, CELF++ (39.04 versus 60.09 seconds on average).

As can be observed in Table 5, CELF is able to provide high quality solutions in small computing time. In order to further analyze these results, we conduct an additional experiment to compare our constructive procedure (i.e., only using the first stage of the GRASP procedure and not considering the application of the local search) with CELF. In this case, the average objective function for C_{ne} is 227.83, which compares favorably to the result achieved with CELF, which is 208.33. In both cases, the computing time is approximately 6 seconds.

Analyzing the computing time required for each algorithm, we can clearly see that CELF and CELF++, as completely greedy approaches, are not really affected by increasing the size of the seed set. On the contrary, the computing time required for PSO and GRASP is affected by the size of the seed set since larger k-values lead the local improvement method to perform a larger number of iterations. However, if we take a closer vision of the results obtained with GRASP, we can conclude that this increase in the number of iterations and, therefore, in the computing time allows the algorithm to reach better solutions. In the case of PSO, the increase of computing time is even much more noticeable but it does not usually result in better solutions, suggesting that the PSO algorithm is particularly suitable for solving small size instances.

In order to validate these results, we have conducted a non-parametric Friedman test for ranking all the compared algorithms. The p-value obtained, smaller than 0.0005, confirms that there are statistically significant differences among the algorithms. The algorithms sorted by ranking are GRASP

(1.00), CELF++ (2.44), CELF (2.79), and PSO (3.77). We finally perform the well-known non-parametric Wilcoxon statistical test for pairwise comparisons, which answers the question: do the solutions generated by both algorithms represent two different populations? The resulting *p*-value smaller than 0.0005 when comparing GRASP with each other algorithm confirms the superiority of the proposed GRASP algorithm. Therefore, GRASP emerges as one of the most competitive algorithms for the SNIMP, being able to reach high quality solutions in small computing time.

5 Conclusions

In this paper a quick GRASP algorithm for solving the SNIMP has been presented. Two constructive procedures are proposed, with the one based on the two-step neighborhood being more competitive one than that based on the clustering coefficient. Furthermore, the idea of using local information allows the algorithm to construct a complete solution in a small computing time. Then, a local search based on swap moves is presented. Since an exhaustive exploration of the search space is not suitable for this problem, we propose an intelligent neighborhood exploration strategy which limits the region of the search space to be explored, focusing on the most promising areas. This rationale leads us to provide high quality solutions in reasonable computing time, even for the largest instances derived from real-world SNs commonly considered in the SNIMP area. Since the intelligent neighborhood exploration strategy is parameterized, if the computing time is not a relevant factor, the region explored can be easily extended to find better solutions, thus increasing the required computational effort. This fact makes the proposed GRASP algorithm highly scalable. The results obtained are supported by Friedman test and then the pairwise Wilcoxon test, confirming the superiority of the proposal with respect to the classical and state-of-the-art solution procedures in the area.

In our future work, we plan to study the adaptation of techniques developed in this work to influence minimization problems. This adaptation can be useful for minimizing the impact of fake news and monitor those users which can eventually transmit misinformation through the network (Wei-Neng et al. 2019; Wang et al. 2017).

Acknowledgements This research was funded by "Ministerio de Ciencia, Innovación y Universidades" under grant ref. PGC2018-095322-B-C22, "Comunidad de Madrid" and "Fondos Estructurales" of European Union with grant refs. S2018/TCS-4566, Y2018/EMT-5062. It is also supported by the Spanish Agencia Estatal de Investigación, the Andalusian Government, the University of Granada, and European Regional Development Funds (ERDF) under grants EXASOCO (PGC2018-101216-B-I00) and AIMAR (A-TIC-284-UGR18).

Funding Funding for open access charge: Universidad de Granada / CBUA.

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

References

Banerjee S, Jenamani M, Pratihar DK (2020) A survey on influence maximization in a social network. Knowl Inf Syst 62:3417–3455. https://doi.org/10.1007/s10115-020-01461-4

Barabási, Albert-László, Pósfai Márton (2016) Network science. Cambridge University Press, Cambridge. ISBN: 9781107076266 1107076269, http://barabasi.com/networksciencebook/

Bucur D, Iacca G (2016) Infuence maximization in social networks with genetic algorithms. In: Applications of evolutionary computation. Springer International Publishing, Berlin, pp 379–392. https://doi.org/10.1007/978-3-319-31204-0_25

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2017) Multi-objective evolutionary algorithms for influence maximization in social networks, pp 221–233. https://doi.org/10.1007/978-3-319-55849-3_15

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2018a) Evaluating surrogate models for multi-objective in-fluence maximization in social networks. In: Proceedings of the genetic and evolutionary computation conference companion on GECCO 18. ACM Press. https://doi.org/10.1145/3205651.3208238

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2018b) Improving multi-objective evolutionary in fluence maximization in social networks. In: Applications of evolutionary computation. Springer International Publishing, Berlin, pp 117–124. https://doi.org/10.1007/978-3-319-77538-8_9

Chen W, Yajun W, Siyu Y (2009) Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining-KDD 09. ACM Press. https://doi.org/10.1145/1557019.1557047

Christakis NA, Fowler JH (2009) Connected: the surprising power of our social networks and how they shape our lives. Little, Brown Spark

D'angelo A, Aditya A, Kang-Xing J, Yun-Fang J, Levy K, Oleksandr M, Yishan W (2009) Targeting advertisements in a social network. US Patent App. 12/195321

Feo T, Resende Mauricio AGC (1989) A probabilistic heuristic for a computationally difficult set covering problem. Oper Res Lett 8(2):67–71. https://doi.org/10.1016/0167-6377(89)90002-3

Feo TA, Resende Mauricio GC, Stuart HS (1994) A greedy randomized adaptive search procedure for maximum independent set. Oper Res 42(5):860–878. https://doi.org/10.1287/opre.42.5.860

Gil-Borrás SG, Pardo E, Antonio A-A, Abraham D (2020) GRASP with Variable Neighborhood Descent for the on- line order batching problem. J Glob Optim 2020:1–31
Gong M, Chao S, Chao D, Lijia M, Bo S (2016) An efficient memetic algorithm for influence maximization in social networks. IEEE Comput Intell Mag 11(3):22–33. https://doi.org/10.1109/mci.2016.2572538
Goyal A, Wei L, Laks VSL (2011) CELF++: optimizing the greedy algorithm for influence maximization in social networks. In: Proceedings of the 20th international conference companion on World wide web WWW 11. ACM Press. https://doi.org/10.1145/1963192.1963217
Hansen P, Mladenović N (2006) First vs. best improvement: an empirical study. In: Discrete Applied Mathematics 154.5. IV ALIO/EURO workshop on applied combinatorial optimization, pp 802–817. Issn: 0166-218X. https://doi.org/10.1016/j.dam.2005.05.020, http://www.sciencedirect.com/science/article/pii/S0166218X05003070
Jiaguo L, Guo J, Yang Z, Zhang W, Jocshi A (2014) Improved algorithms of CELF and CELF++ for influence maximization. J Eng Sci Technol Rev 7(3):32–38. https://doi.org/10.25103/jestr.073.05
Kempe D, Jon K, Éva T (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 KDD 03. ACM Press. https://doi.org/10.1145/956750.956769
Kempe D, Jon K, Eva T (2015) Maximizing the spread of influence through a social network. Theory Comput 11(1):105–147. https://doi.org/10.4086/toc.2015.v011a004
Khalil EB, Dilkina B, Song L (2013)‘CuttingEdge: influence minimization in networks. In: Workshop on frontiers of network analysis: methods, models, and applications at NIPS. url: files/papers/CuttingEdge.pdf
Klovdahl AS (1985) Social networks and the spread of infectious diseases: the AIDS example. Soc Sci Med 21(11):1203–1216. https://doi.org/10.1016/0277-9536(85)90269-2
Lamont CCC, Gary B, van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems. Springer, US. https://doi.org/10.1007/978-0-387-36797-2
Lappas T, Evimaria T, Dimitrios G, Heikki M (2010) Finding effectors in social networks. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 10. ACM Press. https://doi.org/10.1145/1835804.1835937
Lawyer G (2015) Understanding the influence of all nodes in a network. Sci Rep 5:1. https://doi.org/10.1038/srep08665
Lee J-R, Chung C-W (2015) A Query approach for influence maximization on specific users in social networks. IEEE Trans Knowl Data Eng 27(2):340–353. https://doi.org/10.1109/tkde.2014.2330833
Leskovec J, Andreas K, Carlos G, Christos F, Jeanne Van B, Natalie G (2007) Cost-effective outbreak detection in networks. Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 07. ACM Press. https://doi.org/10.1145/1281192.1281239
Li D, Zhi-Ming X, Nilanjan C, Anika G, Katia S, Sheng L (2014) Polarity related influence maximization in signed social networks. PLoS ONE 9(7):e102199. https://doi.org/10.1371/journal.pone.0102199 ((**Ed. by Sergio Gómez**))
Li D, Wang C, Zhang S, Zhou G, Chu D, Chong W (2017) Positive influence maximization in signed social networks based on simulated annealing. Neurocomputing 260:69–78. https://doi.org/10.1016/j.neucom.2017.03.003
Liu B, Gao C, Yifeng Z, Dong X, Yeow MC (2014) Influence spreading path and its application to the time constrained social influence maximization problem and beyond. IEEE Trans Knowl Data Eng 26(8):1904–1917. https://doi.org/10.1109/tkde.2013.106
Liu Y, Xi W, Jurgen K (2019) Framework of evolutionary algorithm for investigation of influential nodes in complex networks. IEEE Trans Evol Comput 23(6):1049–1063. https://doi.org/10.1109/tevc.2019.2901012
Luo C, Kainan C, Xiaolong Z, Zeng D (2014) Time critical disinformation influence minimization in online social networks. In: 2014 IEEE joint intelligence and security informatics conference, pp 68–74
Martí R, Anna M-G, Jesús S-O, Abraham D (2018) Tabu search for the dynamic Bipartite Drawing Problem. Comput Oper Res 91:1–12 ((**issn: 0305-0548**))
Nguyen H, Zheng R (2013) On budgeted influence maximization in social networks. IEEE J Sel Areas Commun 31(6):1084–1094. https://doi.org/10.1109/jsac.2013.130610
Nguyen Hung T, Thai My T, Dinh Thang N (2016) Stop-and-stare: optimal sampling algorithms for viral marketing in billion-scale net- works. In: Proceedings of the 2016 international conference on management of data. SIGMOD ’16. San Francisco, California, USA: association for computing machinery, pp 695–710. isbn: 9781450335317. https://doi.org/10.1145/2882903.2915207. url: https://doi.org/10.1145/2882903.2915207
Ok J, Youngmi J, Jinwoo S, Yung Y (2016) On maximizing diffusion speed over social networks with strategic users. IEEE/ACM Trans Netw 24(6):3798–3811. https://doi.org/10.1109/tnet.2016.2556719
Peng S, Aimin Y, Lihong C, Shui Y, Dongqing X (2017) Social influence modeling using information theory in mobile social networks. Inf Sci 379:146–159. https://doi.org/10.1016/j.ins.2016.08.023
Pérez-Peló S, Jesús S-O, Abraham D (2020) Finding weaknesses in networks using greedy randomized adaptive search procedure and path relinking. Expert Syst 2020:e12540
Pérez-Peló S, Jesús S-O, Raúl M-S, Abraham D (2019) On the analysis of the influence of the evaluation metric in community detection over social networks. Electronics 8(1):23
Qipeng Y, Shi R, Zhou C, Wang P, Guo L (2015) Topic-aware social influence minimization. In: Proceedings of the 24th International Conference on World Wide Web. WWW ’15 Companion. Florence, Italy: Association for Computing Machinery, pp 139–140. ISBN: 9781450334730. https://doi.org/10.1145/2740908.2742767
Resende M, Celso GC, Ribeiro C (2013) GRASP: greedy randomized adaptive search procedures. In: Search methodologies. Springer US, pp 287–312. https://doi.org/10.1007/978-1-4614-6940-7_11
Resende M, Rafael-Martí GC, Micael G, Abraham D (2010) GRASP and path relinking for the max-min diversity problem. Comput Oper Res 37(3):498–508. https://doi.org/10.1016/j.cor.2008.05.011
Reza Z, Abbasi MA, Liu H (2014) Social media mining: an introduction. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9781139088510
Richardson M, Rakesh A, Pedro D (2003) Trust management for the semantic web. In: Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 351–368. https://doi.org/10.1007/978-3-540-39718-2_23
Robles JF, Manuel C, Oscar C (2020) Evolutionary multiobjective optimization to target social network influentials in viral marketing. In: Expert systems with applications, vol 147, p 113183. ISSN: 0957-4174. https://doi.org/10.1016/j.eswa.2020.113183, http://www.sciencedirect.com/science/article/pii/S0957417420300099
Salavati C, Abdollahpouri A (2019) Identifying influential nodes based on ant colony optimization to maximize profit in social networks. Swarm Evol Comput 51:100614. https://doi.org/10.1016/j.swevo.2019.100614
Samadi M, Nagi R, Semenov A, Nikolaev A (2018) Seed activation scheduling for influence maximization in social networks. Omega 77:96–114. https://doi.org/10.1016/j.omega.2017.06.002

Sen P, Lev M, José SA, Zheng ZA, Makse H (2014) Searching for superspreaders of information in real-world social media. Sci Rep 4:1. https://doi.org/10.1038/srep05547

Şimşek A, Kara R (2018) Using swarm intelligence algorithms to detect influential individuals for influence maximization in social networks. Expert Syst Appl 114:224–236. https://doi.org/10.1016/j.eswa.2018.07.038

Song G, Xiabing Z, Yu W, Xie K (2015) Influence maximization on large-scale mobile social network: a divide- and-conquer method. IEEE Trans Parallel Distrib Syst 26(5):1379–1392. https://doi.org/10.1109/tpds.2014.2320515

Wasserman S, Faust K (1994) Social network analysis. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511815478

Tong G, Weili W, Shaojie T, Ding-Zhu D (2017) Adaptive influence maximization in dynamic social networks. IEEE/ACM Trans Netw 25(1):112–125. https://doi.org/10.1109/tnet.2016.2563397

Tong GA, Shasha L, Weili W, Ding-Zhu D (2016) Effector detection in social networks. IEEE Trans Comput Soc Syst 3(4):151–163. https://doi.org/10.1109/tcss.2016.2627811

Wang B, Ge C, Luoyi F, Li S, Xinbing W (2017) Drimux: dynamic rumor influence minimization with user experience in social networks. IEEE Trans Knowl Data Eng 29(10):2168–2181

Watts DJ, Strogatz SH (1998) Collective dynamics of small-world networks. Nature 393(6684):440–442. https://doi.org/10.1038/30918

Wei-Neng C, Tan D-Z, Yang Q, Gu T, Zhang J (2019) Ant colony optimization for the control of pollutant spreading on social networks. IEEE Trans Cybern 50(9):4053–4065

Xinjue W, Deng K, Li J, Yu JX, Jensen SC, Yang X (2018) Targeted influence minimization in social networks. In: Phung D, Tseng VS, Webb GI, Ho B, Ganji M, Rashidi L (eds) Advances in knowledge discovery and data mining. Springer International Publishing, Cham, pp 689– 700. ISBN: 978-3-319-93040-4

Yang W-S, Weng S-X (2012) Application of the ant colony optimization algorithm to competitive viral marketing. In: Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 1–8. https://doi.org/10.1007/978-3-642-30448-4_1

Zhang H, Dung TN, Huiling Z, My TT (2016) Least cost influence maximization across multiple social networks. IEEE/ACM Trans Netw 24(2):929–939. https://doi.org/10.1109/tnet.2015.2394793

Zhang K, Haifeng D, Feldman MW (2017) Maximizing influence in a social network: improved results using a genetic algorithm. Phys A 478:20–30. https://doi.org/10.1016/j.physa.2017.02.067