New metaheuristic algorithms for the analysis of the user influence in social networks

Isaac Lozano-Osorio[1][0000−0002−2608−8464], Jesús Sánchez-Oro[1][000−0003−1702−4941], Oscar Cordón[2][0000−0001−5112−5629], and Abraham Duarte[1][0000−0002−4532−3124]

[1] Universidad Rey Juan Carlos, Madrid, Spain
{isaac.lozano,jesus.sanchezoro,abraham.duarte}@urjc.es
[2] Universidad de Granada, Granada, Spain
ocordon@decsai.ugr.es

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 are usually known as Social Network Influence Maximization (SNIM) problems, whose goal is to find the most influential users (commonly known as seeds) in the social network, simulating an influence diffusion model. Since SNIM is known to be an $\mathcal{NP}$-hard problem, and most of the networks to analyze are considerably large, this problem have become a real challenge for the scientific community. Different works have attempted to solve it by means of heuristic or metaheuristic approaches, such as evolutionary algorithms based on simulations of the spreading mechanism, but they are not able to solve real-world large-scale networks due to their limitations in computing time. The main drawback of this optimization problem lies in the computational effort required to evaluate a solution. Since the infection of a node is evaluated in a probabilistic manner, the objective function value requires from a Monte Carlo simulation to see the robustness of the method, resulting in a computationally complex process. The current proposal tries to overcome this limitations by considering a metaheuristic algorithm based on the Greedy Randomized Adaptive Search Procedure (GRASP) framework. The construction phase 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 local improvement involves a surrogate-assisted-swap local search to find the most influential users based on swap moves. Experiments performed on 6 well-known social networks 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, Graph theory, Viral marketing, GRASP.

1 Introduction

Nowadays, social networks (SNs) are daily used by millions of users, while the number of users continues to grow exponentially. This growth is extended to the amount of behavioral data and, therefore, all classical network-related problems are becoming computationally harder. SN can be defined as the representation of social interactions that can be used to study the propagation of ideas, detection of groups, finding weaknesses, social bond dynamics, disease propagation, viral marketing, or advertisement, among others [1, 2, 3, 4, 5]. Algorithmically, a social network is 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 $u, v \in V$ which indicates that user u can transmit information to (or infect) user v.

Kempe et al. [6] originally formalized the influence model to analyze how the information is transmitted among the users of a SN. Given a network with n nodes where the edges represent the spreading or propagation process on that network, the task is to choose a seed 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 problem was initially formulated in [7] and it was later proven $\mathcal{NP}$-hard for most Influence Diffusion Models (IDMs) [8]. 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.

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. Several IDMs can be found in the literature but the most extended one is the Independent Cascade Model [6], a probabilistic IDM where the edge (u, v) has a weight P representing the probability with which u is able to influence v.

This work presents a novel metaheuristic approach for dealing with this problem, allowing us to find high quality solutions in the context of SNIM in a short computing times. 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.

The remainder of the work is structured as follows. Section 2 formalizes the problem and reviews the related literature. The proposed approach is described within detail in Section 3. Section 4 presents the experimental results considering a public dataset which has been previously used for this task. The results obtained are compared against the most popular algorithms for SNIM and they clearly demonstrate the efficacy of the proposed methodology. Finally, Section 5 draws some conclusions derived from this research.

2 State of the art

In this section, we introduce some related work about SNIM and IDMs as well as we briefly survey the existing methods for solving this problem, based on either heuristics or computational intelligence algorithms.

Richardson and Domingos [7] initially formulated the problem of selecting target nodes in SNs. However, Kempe et al. [6] tried to solve the SNIM problem, formulating it as a discrete optimization problem. It has been shown that computation of the influence spread of a given node set is $\mathcal{NP}$-hard [8]. Kempe et al. [6] proposed a greedy algorithm called *greedy hill-climbing algorithm* with an approximation of $1 - 1/e$. Besides, they also proved that this greedy algorithm can obtain a solution within 63% of optimality under three commonly used IDMs as the Independent Cascade Model (ICM), the Weighted Cascade Model (WC), and the Linear Threshold model (LT). ICM is one of the most popular IDMs and it is the one used in this contribution. It considers the transmission of the influence through the network in a tree-like fashion, where the seed nodes are the roots.

The most extended way of evaluating the spread in ICM is 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 the network given a seed set. The algorithm performs a number of predefined iterations IT, finding in each iteration which are the nodes that receives the information given a seed set S (steps $2 - 18$). 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 infected (steps 5-16). In each iteration, the new infected nodes are traversed in order to check if they are able to transmit information to non-infected nodes (steps 8-12). For each non-infected node adjacent to one of the newly infected nodes, a random number is generated. If this number is smaller than the probability of infection P, then it becomes infected (steps 9-11). At the end of the iteration, 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 during the predefined number of iterations (step 19). This value, called *influence spread*, is considered as the objective function to be optimized by the greedy algorithm. That is, the seed set maximizing the spread value would compose the optimal solution to the problem. Notice that, as infection is an stochastic process, the ICM must be run several times to achieve an appropriate estimation by the mean of the variable, thus resulting in a Monte Carlo simulation.

As a consequence of this computationally expensive drawback, Kempe et al [6] also proposed several greedy heuristics based on social network analysis measures such as degree and closeness centrality [9]. When the considered measure 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. [10] introduced the cost-effective lazy forward (*CELF*) selection which exploited the submodularity property to significantly reduce the

Algorithm 1 $ICM(G = (V, E), S, P, IT)$

```
1: spread ← ∅
2: for i ∈ 1 ... IT do
3:     A* ← S
4:     A ← S
5:     while A ≠ ∅ do
6:         B ← ∅
7:         for v ∈ A do
8:             for (u, v) ∈ E do
9:                 if rnd(0, 1) ≤ p then
10:                    B ← B ∪ {u}
11:                end if
12:            end for
13:        end for
14:        A* ← A* ∪ B
15:        A ← B
16:    end while
17:    spread ← spread + |A*|
18: end for
19: return spread/IT
```

run time of the greedy hill-climbing algorithm, becoming over 700 times faster than greedy hill-climbing algorithm. 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, Wang, and Yang [11] 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 count those reached by the already selected seed nodes to develop a better estimation of the total spread.

In [12] a new algorithm called *CELF++* was proposed with the aim of improving the efficiency of the original CELF. According to the authors it is 35-55% faster than *CELF*. However, further studies [13] showed that *CELF++* is much slower than *CELF*, which is different from the results reported in [12]. In this work algorithm *CELF++* was carefully re-implemented following the instructions of [12] and tested with different instances, confirming the results published in [13].

A large number of works have been developed in the area since those first proposals [14]. Different kinds of heuristic and metaheuristic algorithms have been considered to solve the SNIM problem. Table 1 shows the acronyms of the methods reported in Table 2.

Analyzing previous studies we can conclude that more complex metaheuristic approaches usually result in better solutions than simple greedy approaches. Yang et al. [37] proposed an Ant Colony Optimization (ACO) algorithm based on a parameterized probabilistic model to address the SNIM problem. They

Table 1: Long terms acronym

Definition	Acronym
Independent Cascade Model	ICM
Independent Poisson Clock	IPC
Dynamic Programming	DP
Latency Aware	LA
Linear Threshold	LT
Polarity-Related	PR
Simulated Annealing	SA
Suspected Infected	SI
Mixed Integer Programming	MIP
Partial Parallel Cascade	PPX
Multi-Objective Evolutionary Algorithm	MOEA
SIR epidemic spreading model [15]	SIR

Table 2: State of the art

Study	Algorithm	Model
(Kempe et al., 2003)[6]	Greedy	ICM-LT
(Jiaguo et al., 2014)[13]	Greedy	ICM
(Goyal et al., 2011)[12]	Greedy	ICM
(Tong et al., 2016)[16]	Greedy	ICM
(Song et al., 2017)[17]	Greedy	ICM
(Ok et al., 2016)[18]	Greedy	IPC
(Lappas et al., 2010)[19]	DP	ICM
(Nguyen, & Zheng, 2013)[20]	Greedy	ICM
(Liu et al., 2014)[21]	Greedy	LA-ICM
(Song et al., 2015)[22]	Greedy	ICM
(Lee & Chung, 2015)[23]	Greedy	ICM
(Zhang et al., 2016)[24]	Greedy	LT-ICM
(Gong et al., 2016)[25]	Particle Swarm	ICM
(Chen et al., 2009)[11]	Greedy	ICM
(Zhang et al., 2017)[26]	Genetic	LT
(Li et al., 2014)[27]	Greedy	PR-ICM
(Li et al., 2017)[28]	SA	PR-ICM
(Peng et al., 2017)[29]	Greedy	SI
(Samadi et al., 2018)[30]	MIP	PPC
(Bucur et al., 2018)[31]	MOEA	ICM
(Bucur et al., 2017)[32]	MOEA	ICM
(Bucur et al., 2016)[33]	Genetic	ICM
(Liu et al., 2019)[34]	Evolutionary	SIR
(Salavati et al., 2019)[35]	Ant Colony	ICM
(Bucur et al., 2018)[36]	MOEA	ICM

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 [38] 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 [32], an Evolutionary Multiobjective Optimization (EMO) algorithm [39] was proposed for SNIM, where the two considered objectives were (i) maximizing the influence of a seed set, and (ii) minimizing the number of nodes in the seed set.

As said, some heuristics have been proposed as time-saving solutions for greedy decisions: random, degree, and centrality [6]. 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 social network analysis [9]. Degree centrality heuristic usually produces less accurate results to the SNIM problem. Based on the high-degree heuristic, another heuristic, targets the SNIM problem by taking into account prior knowledge of the node's neighbors [11].

The aforementioned metaheuristic methods are able to obtain good results but require large computational times. Our proposal considers a balanced method based on the ICM. With the aim of reducing the required computing time, we consider the GRASP methodology, combining good heuristic solutions that are fastly generated and an efficient local search method which minimizes the number of Monte Carlo evaluations to further improve the initial solution.

3 Greedy Randomized Adaptive Search Procedure

Greedy Randomized Adaptive Search Procedure (GRASP) is a metaheuristic developed in the late 1980s [40] and formally introduced in Feo et al. [41]. We refer the reader to [42, 43] for a complete survey of the last advances in this methodology. GRASP is a multi-start framework that consists of two distinct stages: construction and local search. The former is a greedy, randomized, and adaptive method to generate a solution from scratch. The latter is devoted to obtain a local optimum with respect to a certain neighborhood starting from the constructed solution. These two phases are repeated until a termination criterion is met, returning the best solution found.

3.1 Construction phase

The construction phase of GRASP is devoted for generating 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. Algorithm 2 depicts the pseudocode of the proposed constructive procedure.

The method starts from an empty solution S (step 1), creating the candidate list, CL, with the complete set of vertices V (step 2). Then, the constructive method iteratively adds new elements to the solution until it becomes feasible (steps 3-11). In each iteration, the minimum and maximum value of the greedy heuristic is evaluated (steps 4-5). Then, a threshold μ is calculated (step 6). This

threshold is required for creating the Restricted Candidate List (RCL) with the most promising nodes (step 7). Finally, the next node is selected at random from the RCL (step 8), including it in the solution (step 9) and updating the CL (step 10). The method ends returning the constructed solution S (step 12).

Algorithm 2 $GRASP(G = (V, E))$

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

The greedy heuristic function (used in steps 4-5) is one of the features that will improve the GRASP solution. In the context of this problem, we propose different greedy functions to generate the initial solution.

The first one (**ALG-NEIGH**) is a heuristic based on the first and second degree neighbors of a given user. Given a user u, this method calculates the sum of out degree of u and its neighbors. Let us define $d(A)$ as the degree of node A. Then, in the graph depicted in Figure 1, the heuristic value for node A is $d(A) + d(B) = 1 + 4 = 5$, while the heuristic value for node B is 11 due to $d(B) + d(A) + d(C) + d(E) + d(F) = 4 + 1 + 2 + 2 + 2 = 11$. Then, if we represent, for each node v, the heuristic value $h(v)$, in a vector where the first position corresponds to node A, the second one to node B, and so on, it results in $[5, 11, 6, 2, 6, 6, 2, 2]$. Then the node with the highest value is chosen, the remaining nodes are updated by subtracting the selected node degree to his neighbors and for all nodes subtracting the number of neighbors that has an edge with selected node. In the example, the user selected is B, because it presents the largest value in the vector, and the nodes degrees are recomputed as $[1, X, 2, 1, 1, 1, 2, 2]$. For example the update of node F is computed as $|\{h(F)\}| - |\{d(B), E\}| = 6 - 4 - 1 = 1$.

The second greedy criterion considers the clustering coefficient as heuristic value (**ALG-CLUS**). The clustering coefficient is the measure of the degree to which nodes in a graph tend to cluster together. To speed up its computation, every value is precalculated before the algorithm's run. All the information related to the implementation of the clustering coefficient can be found in [44].

Closeness centrality is another way of detecting nodes that are able to spread information. The closeness centrality of a node measures its average distance to

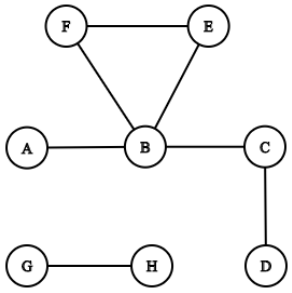


Fig. 1: Example graph with 8 users and 7 relations among them to show the heuristic function proposed to evaluate each user.

all the other nodes. Nodes with large closeness scores have the shortest distances to all other nodes. The Closeness of a node v is evaluated as $C(v) = \frac{n-1}{\sum_{u \in V} d(u,v)}$. The algorithm calculates the sum of Hamiltonian distances to all other nodes, based on calculating the shortest paths between every pair of nodes. The resulting sum is then averaged and inverted to determine the closeness centrality score for that node.

An efficient method to get the Closeness (represented by **ALG-CLOSS**) value is to precalculate the shortest path distances for every pair of nodes and save it in cache. That can be done with a Breadth First Search over all nodes [45]. Once the minimum distance over all nodes is obtained, we have to compute its average. The main drawback is the memory needed to store this information in real-life networks, requiring an extra matrix of n^2 elements.

3.2 Local Search Phase

The second phase of GRASP consists of improving the solution generated by the constructive procedure with the aim of reaching a local optimum. This phase can be accomplished by using simple local search procedures or more complex heuristics. Due to the complexity of the problem under consideration, we propose a local search procedure.

In order to properly define a local search, we first need to introduce the neighborhood considered. The neighborhood of a solution can be defined as the set of new solutions that can be reached by performing a single move over the original solution. Therefore, a move must be defined in order to present the neighborhood. In the context of this problem, the swap move is considered, which replaces a selected seed node with a non-selected one. This swap move is formally defined as:

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

Then, the neighborhood N_s of a given solution S is defined as the set of solutions that can be reached by performing a single swap move to S. More formally,

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

Once the neighborhood has been defined, we finally need to clarify in which order it is explored. Specifically, we propose to randomly traverse the neighborhood performing the first move that leads to a better solution (First Improvement).

Due to the vast size of the resulting neighborhood, $k \times (n - k)$, a surrogate search is presented with the aim of reducing the number of solutions explored within each neighborhood. The search space explored is limited by reducing the percentage of nodes considered for the swap move.

Two different proposals are used to select the number of neighboring solutions to be explored for each solution. The first idea is to limit the maximum number of solutions to be explored, X. Following this approach, the local search will explore $min(|N_s(S)|, X)$ neighbor solution. Notice that the larger the value of X, the better the quality of the solutions but also the higher the computing time required to reach them.This approach is named as *Minimum N Values* (MNV). The second approach takes into account a certain percentage Y of neighbors to explore to ease the scalability of the process, defined as $|N_s(S)| * Y$, and it is named as *Percentage of Candidates Values (PCV)*. Notice that the number of neighbors explored directly depends on the network size. The results of both proposals are provided in next section.

4 Experiments

This section describes the experimental setup designed to validate the proposal. Relevant information about the instances used is collected in Table 3. All of them are publicly available in Stanford Network Analysis Project (SNAP): `https://snap.stanford.edu/`. All the experiments have been performed in a computer with an AMD Ryzen 5 3600 (3,60 GHz) and 16GB RAM.

Table 3: Instances

Instance	***Nodes***	***Edges***	***Diameter***
p2p-Gnutella31	62586	147892	11
ca-AstroPh	18772	198110	14
ca-CondMat	23133	93497	14
cit-HepPh	34546	421578	12
email-Enron	36692	183831	11
email-EuAll	265214	420045	14

First of all, it is important to show the values used for the ICM algorithm with Monte Carlo and the number of iterations performed to get the best value with GRASP. In all the experiments, GRASP constructs 100 solutions and ICM runs for 100 iterations (IT) with a probability of infection (P) of 0.01%. These parameter values are the most extended ones in the literature. The number of seed nodes K to conform a solution are selected in the range $K = \{10, 20, 30, 40, 50\}$, thus obtaining $6 \times 5 = 30$ different problem instances (6 networks and 5 seed set sizes).

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

All the experiments report the following metrics: OF, the average objective function value; Time (s), the average computing time required by the algorithm in seconds; Dev(%), the average deviation with respect to the best value found in the experiment; and # Best, the number of times that the algorithm matches the best solution.

The preliminary experimentation has been performed with a small set of instances (CA-AstroPh and CA-CondMat with $K = \{10, 20\}$; and Email-Enron and Email-EuAll with $K = \{30, 40, 50\}$) to avoid overfitting. This selection comprehends a 33.33% of the global set and provide enough variability in instances and values of K. The results obtained in this preliminary experimentation are shown in Table 4. As it can be seen, **ALG-NEIGH** provides the best results in both quality and computing time, so it is the selected greedy function to use in GRASP constructive.

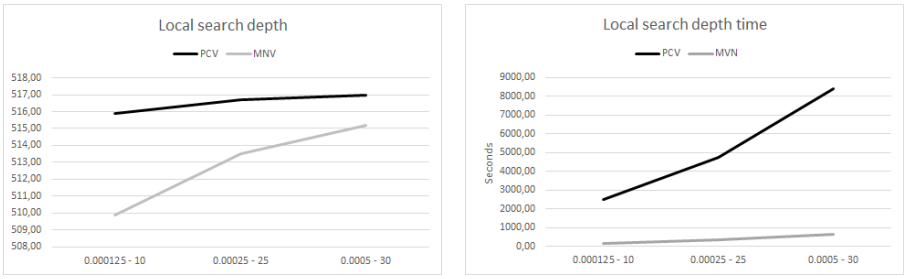
Table 4: Constructive algorithms results

	OF	$Time(s)$	$Dev(\%)$	$Best$
$ALG-HEUR$	490.80	14.69	0.00	10/10
$ALG-CLUS$	356.10	81.14	34.33	0/10
$ALG-CLOSS*$	459.57	5.68	43.88	0/10

* The algorithm did not finish due to memory limits.

The second experiment is devoted to test the impact of the surrogate local search to select the X and Y values as shown in the previous section with the same instances used in the last experimentation. The X values tested are {10, 25, 30} in MNV and {0.000125, 0.00025, 0.0005} for PCV. The Y value denote the average number of nodes influenced. In view of the results obtained, the better the values, the larger the computing time required. Figure 2a compares the quality using the average number of influenced nodes while in Figure 2b shows the run time taken. Analyzing Figure 2a it is showed that PCV achieves better quality than MNV. The average number of influenced nodes reach the

largest values when considering the largest parameter values but the MNV is faster than the PCV in all the cases (see Figure 2b). Analyzing the results of the PCV we found a major drawback. If the same percentage Y is used in a large and also in a small social network, it could require long times for the large network if it is a large value and it can deteriorate the quality of the solutions generated in a small social network if a small value is chosen. Since MNV presents similar quality in the last experiment and the computing time for PCV is considerably larger than the one for MNV, we select MNV to be used in the local search phase.



(a) (b)

Fig. 2: Quality (left) and time (right) comparison for PCV and MNV

The final experiment is intended to evaluate the quality of the best proposed GRASP design, conformed with **ALG-NEIGH** in constructive phase and the **MNV** formula in local search phase, when comparing it with the best method found in the state of the art, CELF. Table 5 shows the results of the comparison. Notice that the run time is lower for GRASP in every case but K=50 where it is higher but at the expense of obtaining a more accurate solution than *CELF*. In all the executions, GRASP performs equal or better than CELF maintaining competitive running times. The last column shows the number of instances where each algorithm reaches the best solution.

Figure 3a graphically illustrates that the superiority of GRASP become more evident when the seed set size increases, but we can also see in Figure 3b how the run time also grows.

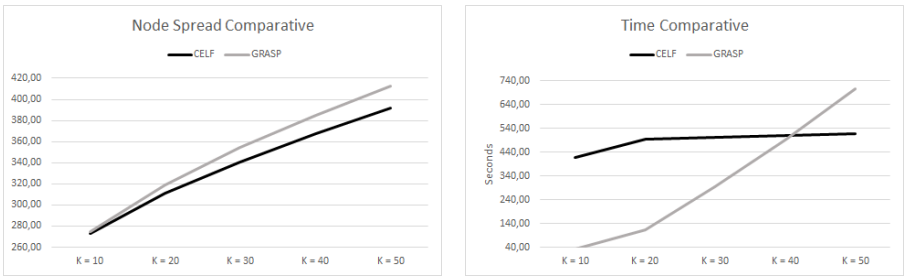
In order to confirm that there are statistically significant differences between both algorithms, we have performed the well-known non-parametric Wilcoxon test. The resulting p-value smaller than 0.001 confirms these differences and, therefore, the superiority of the proposal.

5 Conclusions

In this work a GRASP algorithm has been proposed for the SNIM problem. Different heuristic functions and surrogate local search designs were tested to

Table 5: GRASP vs CELF

	K	OF	Time (s)	Dev(%)	#Best
CELF	10	272.83	416.90	0.23	5/6
GRASP	10	**274.33**	**31.84**	**0.12**	4/6
CELF	20	311.00	494.18	1.02	4/6
GRASP	20	**318.83**	**115.42**	**0.00**	**6/6**
CELF	30	341.00	502.78	2.03	3/6
GRASP	30	**354.67**	**294.06**	**0.00**	**6/6**
CELF	40	367.17	509.34	2.53	3/6
GRASP	40	**384.83**	**489.75**	**0.00**	**6/6**
CELF	50	391.67	516.21	2.74	3/6
GRASP	50	**412.17**	707.71	**0.08**	**5/6**



(a) (b)

Fig. 3: Quality (left) and time (right) comparison for GRASP and CELF.

reduce the number of solutions to be explored and get better solutions in terms of both quality and computing time. The heuristic function provides flexibility in the GRASP method design in order to reach a trade-off between the quality and the computing time. *GRASP* offers very competitive results with respect to CELF, drastically reducing the computing time while providing better solutions on average.

Therefore, it can be concluded that a careful algorithmic design is necessary in SNIM in order to be able to deal with large social networks without requiring vast computing times. The surrogate local search presented in this work is able to balance quality and computing time by means of a single parameter, which makes it adequate for most social networks.

Acknowledgment

This work has been partially supported by the "Ministerio de Ciencia, Innovación y Universidades" under grant ref. PGC2018-095322-B-C22 and "Comunidad de Madrid" and "Fondos Estructurales" of European Union with grants ref. S2018/TCS-4566 and Y2018/EMT-5062.

References

[1] Stephen F. King and Thomas F. Burgess. "Understanding success and failure in customer relationship management". In: *Industrial Marketing Management* 37.4 (June 2008), pp. 421–431. DOI: 10.1016/j.indmarman.2007.02.005.

[2] Adam D'angelo et al. *Targeting advertisements in a social network*. US Patent App. 12/195,321. Mar. 2009.

[3] Alden S. Klovdahl. "Social networks and the spread of infectious diseases: The AIDS example". In: *Social Science & Medicine* 21.11 (Jan. 1985), pp. 1203–1216. DOI: 10.1016/0277-9536(85)90269-2.

[4] Sergio Pérez-Peló et al. "On the Analysis of the Influence of the Evaluation Metric in Community Detection over Social Networks". In: *Electronics* 8.1 (2019), p. 23.

[5] Sergio Pérez-Peló, Jesús Sánchez-Oro, and Abraham Duarte. "Finding weaknesses in networks using Greedy Randomized Adaptive Search Procedure and Path Relinking". In: *Expert Systems* (2020), e12540.

[6] David Kempe, Jon Kleinberg, and Éva Tardos. "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, 2003. DOI: 10.1145/956750.956769.

[7] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. "Trust Management for the Semantic Web". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2003, pp. 351–368. DOI: 10.1007/978-3-540-39718-2_23.

[8] David Kempe, Jon Kleinberg, and Eva Tardos. "Maximizing the Spread of Influence through a Social Network". In: *Theory of Computing* 11.1 (2015), pp. 105–147. DOI: 10.4086/toc.2015.v011a004.

[9] Stanley Wasserman and Katherine Faust. *Social Network Analysis*. Cambridge University Press, Nov. 1994. DOI: 10.1017/cbo9780511815478.

[10] Jure Leskovec et al. "Cost-effective outbreak detection in networks". In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*. ACM Press, 2007. DOI: 10.1145/1281192.1281239.

[11] Wei Chen, Yajun Wang, and Siyu Yang. "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, 2009. DOI: 10.1145/1557019.1557047.

[12] Amit Goyal, Wei Lu, and Laks V.S. Lakshmanan. "CELF++". In: *Proceedings of the 20th international conference companion on World wide web - WWW '11*. ACM Press, 2011. DOI: 10.1145/1963192.1963217.

[13] Jiaguo Lv et al. "Improved Algorithms OF CELF and CELF++ for Influence Maximization". In: *Journal of Engineering Science and Technology Review* 7.3 (Aug. 2014), pp. 32–38. DOI: 10.25103/jestr.073.05.

[14] Aybike ŞİMŞEK and Resul KARA. "Using swarm intelligence algorithms to detect influential individuals for influence maximization in social net-

works". In: *Expert Systems with Applications* 114 (Dec. 2018), pp. 224–236. DOI: 10.1016/j.eswa.2018.07.038.

[15] Linton C. Freeman. "Centrality in social networks conceptual clarification". In: *Social Networks* 1.3 (Jan. 1978), pp. 215–239. DOI: 10.1016/0378-8733(78)90021-7.

[16] Guangmo Amo Tong et al. "Effector Detection in Social Networks". In: *IEEE Transactions on Computational Social Systems* 3.4 (Dec. 2016), pp. 151–163. DOI: 10.1109/tcss.2016.2627811.

[17] Guangmo Tong et al. "Adaptive Influence Maximization in Dynamic Social Networks". In: *IEEE/ACM Transactions on Networking* 25.1 (Feb. 2017), pp. 112–125. DOI: 10.1109/tnet.2016.2563397.

[18] Jungseul Ok et al. "On Maximizing Diffusion Speed Over Social Networks With Strategic Users". In: *IEEE/ACM Transactions on Networking* 24.6 (Dec. 2016), pp. 3798–3811. DOI: 10.1109/tnet.2016.2556719.

[19] Theodoros Lappas et al. "Finding effectors in social networks". In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*. ACM Press, 2010. DOI: 10.1145/1835804.1835937.

[20] Huy Nguyen and Rong Zheng. "On Budgeted Influence Maximization in Social Networks". In: *IEEE Journal on Selected Areas in Communications* 31.6 (June 2013), pp. 1084–1094. DOI: 10.1109/jsac.2013.130610.

[21] Bo Liu et al. "Influence Spreading Path and Its Application to the Time Constrained Social Influence Maximization Problem and Beyond". In: *IEEE Transactions on Knowledge and Data Engineering* 26.8 (Aug. 2014), pp. 1904–1917. DOI: 10.1109/tkde.2013.106.

[22] Guojie Song et al. "Influence Maximization on Large-Scale Mobile Social Network: A Divide-and-Conquer Method". In: *IEEE Transactions on Parallel and Distributed Systems* 26.5 (May 2015), pp. 1379–1392. DOI: 10.1109/tpds.2014.2320515.

[23] Jong-Ryul Lee and Chin-Wan Chung. "A Query Approach for Influence Maximization on Specific Users in Social Networks". In: *IEEE Transactions on Knowledge and Data Engineering* 27.2 (Feb. 2015), pp. 340–353. DOI: 10.1109/tkde.2014.2330833.

[24] Huiyuan Zhang et al. "Least Cost Influence Maximization Across Multiple Social Networks". In: *IEEE/ACM Transactions on Networking* 24.2 (Apr. 2016), pp. 929–939. DOI: 10.1109/tnet.2015.2394793.

[25] Maoguo Gong et al. "An Efficient Memetic Algorithm for Influence Maximization in Social Networks". In: *IEEE Computational Intelligence Magazine* 11.3 (Aug. 2016), pp. 22–33. DOI: 10.1109/mci.2016.2572538.

[26] Kaiqi Zhang, Haifeng Du, and Marcus W. Feldman. "Maximizing influence in a social network: Improved results using a genetic algorithm". In: *Physica A: Statistical Mechanics and its Applications* 478 (July 2017), pp. 20–30. DOI: 10.1016/j.physa.2017.02.067.

[27] Dong Li et al. "Polarity Related Influence Maximization in Signed Social Networks". In: *PLoS ONE* 9.7 (July 2014). Ed. by Sergio Gómez, e102199. DOI: 10.1371/journal.pone.0102199.

[28] Dong Li et al. "Positive influence maximization in signed social networks based on simulated annealing". In: *Neurocomputing* 260 (Oct. 2017), pp. 69–78. DOI: 10.1016/j.neucom.2017.03.003.

[29] Sancheng Peng et al. "Social influence modeling using information theory in mobile social networks". In: *Information Sciences* 379 (Feb. 2017), pp. 146–159. DOI: 10.1016/j.ins.2016.08.023.

[30] Mohammadreza Samadi et al. "Seed activation scheduling for influence maximization in social networks". In: *Omega* 77 (June 2018), pp. 96–114. DOI: 10.1016/j.omega.2017.06.002.

[31] Doina Bucur et al. "Improving Multi-objective Evolutionary Influence Maximization in Social Networks". In: *Applications of Evolutionary Computation*. Springer International Publishing, 2018, pp. 117–124. DOI: 10.1007/978-3-319-77538-8_9.

[32] Doina Bucur et al. "Multi-objective Evolutionary Algorithms for Influence Maximization in Social Networks". In: *Applications of Evolutionary Computation*. Springer International Publishing, 2017, pp. 221–233. DOI: 10.1007/978-3-319-55849-3_15.

[33] Doina Bucur and Giovanni Iacca. "Influence Maximization in Social Networks with Genetic Algorithms". In: *Applications of Evolutionary Computation*. Springer International Publishing, 2016, pp. 379–392. DOI: 10.1007/978-3-319-31204-0_25.

[34] Yang Liu, Xi Wang, and Jurgen Kurths. "Framework of Evolutionary Algorithm for Investigation of Influential Nodes in Complex Networks". In: *IEEE Transactions on Evolutionary Computation* 23.6 (Dec. 2019), pp. 1049–1063. DOI: 10.1109/tevc.2019.2901012.

[35] Chiman Salavati and Alireza Abdollahpouri. "Identifying influential nodes based on ant colony optimization to maximize profit in social networks". In: *Swarm and Evolutionary Computation* 51 (Dec. 2019), p. 100614. DOI: 10.1016/j.swevo.2019.100614.

[36] Doina Bucur et al. "Evaluating surrogate models for multi-objective influence maximization in social networks". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion on - GECCO '18*. ACM Press, 2018. DOI: 10.1145/3205651.3208238.

[37] Wan-Shiou Yang and Shi-Xin Weng. "Application of the Ant Colony Optimization Algorithm to Competitive Viral Marketing". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 1–8. DOI: 10.1007/978-3-642-30448-4_1.

[38] Qing-lai JIANG et al. "Preparation of high active Pt/C cathode electrocatalyst for direct methanol fuel cell by citrate-stabilized method". In: *Transactions of Nonferrous Metals Society of China* 21.1 (Jan. 2011), pp. 127–132. DOI: 10.1016/s1003-6326(11)60688-2.

[39] Gary B. Lamont Carlos Coello Coello and David A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer US, 2007. DOI: 10.1007/978-0-387-36797-2.

[40] Thomas A Feo and Mauricio G.C Resende. "A probabilistic heuristic for a computationally difficult set covering problem". In: *Operations Research Letters* 8.2 (Apr. 1989), pp. 67–71. DOI: 10.1016/0167-6377(89)90002-3.

[41] Thomas A. Feo, Mauricio G. C. Resende, and Stuart H. Smith. "A Greedy Randomized Adaptive Search Procedure for Maximum Independent Set". In: *Operations Research* 42.5 (Oct. 1994), pp. 860–878. DOI: 10.1287/opre.42.5.860.

[42] M.G.C. Resende et al. "GRASP and path relinking for the max–min diversity problem". In: *Computers & Operations Research* 37.3 (Mar. 2010), pp. 498–508. DOI: 10.1016/j.cor.2008.05.011.

[43] Mauricio G. C. Resende and Celso C. Ribeiro. "GRASP: Greedy Randomized Adaptive Search Procedures". In: *Search Methodologies*. Springer US, July 2013, pp. 287–312. DOI: 10.1007/978-1-4614-6940-7_11.

[44] Duncan J. Watts and Steven H. Strogatz. "Collective dynamics of 'small-world' networks". In: *Nature* 393.6684 (June 1998), pp. 440–442. DOI: 10.1038/30918.

[45] E. F. MOORE. "The shortest path through a maze". In: *Proc. Int. Symp. Switching Theory, 1959* (1959), pp. 285–292. URL: https://ci.nii.ac.jp/naid/10015375086/en/.