Social Network Analysis and Mining (2024) 14:165
https://doi.org/10.1007/s13278-024-01336-4

ORIGINAL ARTICLE

A variable neighborhood search approach for the adaptive multi round influence maximization problem

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

Received: 4 January 2024 / Revised: 7 August 2024 / Accepted: 8 August 2024

Abstract
Social Networks have been in continuous growing during the last decades. The huge amount of information and applications has led to an increase in the interest of scientists and practitioners in the study of problems related to the influence in Social Networks. Some of the wide variety of real-world applications in this area are viral marketing, disease analysis, rumor detection, public opinion, among others. In this paper, the Adaptive Multi Round Influence Maximization problem is studied, in which the influence of a set of selected users (seed set) is propagated in multiple rounds independently, with the possibility of selecting different seed sets in each round. Therefore, seed sets can be adaptively selected based on the propagation results in the previous rounds. Since each node is activated with a certain probability, the total number of activated nodes must be calculated through an Influence Diffusion Model (IDM), which results in a rather computationally demanding method. In this research, the Independent Cascade Model is considered, which is one of the most extended IDMs, and also the one used in the best previous method. Practitioners highlight the relevance of designing an algorithm capable of efficiently solving the problem. In this research, the problem is addressed by considering the Variable Neighborhood Search methodology, proposing a novel constructive method that relies on independent probability based on events, and an intelligent local search method. Our best algorithm is compared with the state-of-the-art method, named AdaIMM, to analyze the performance of the proposal. The obtained results show the superiority of the proposal in both quality (influence spread) and computing time, obtaining the best solution in all the 40 instances considered requiring half of the computing time than the best previous approach (28 s vs. 53 s). Additionally, the best previous method presents an average deviation of 24.23%. These results are further confirmed by conducting non-parametric statistic tests.

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

1 Introduction

Social Networks (SNs) have become an integral part of people's lives, and, as a result, a large amount of data is generated daily. This growth is extended to the amount of behavioral data, and, therefore, all classical network-related problems are becoming computationally harder. Social Networks information has significant research value for marketing, rumor detection, public opinion analysis, ideas propagation, social bond dynamics, disease propagation, viral marketing, "word-of-mouth" or advertisement, among others Klovdahl (1985); King and Burgess (2008); D'angelo et al. (2009); Berger (2014). The concept of Influence Maximization is an important aspect of research on the use of social network data, which aims to locate a group of nodes to maximize their influence under a specific network propagation model. Thus, successfully solving this problem allows the decision maker to identify the best way to propagate information about products and/or services. For instance, companies can create a wide advertising cascade on social media by selecting individuals to use their products for free, and they must choose users who have the largest effect for accepting this product and recommending it to others. Therefore, the Influence Maximization Problem is devoted

✉ Jesús Sánchez-Oro
jesus.sanchezoro@urjc.es

Isaac Lozano-Osorio
isaac.lozano@urjc.es

Abraham Duarte
abraham.duarte@urjc.es

[1] Department of Computer Science and Statistics, Universidad Rey Juan Carlos, C/ Tulipán s/n., Móstoles 28933, Madrid, Spain

to maximizing the influence of positive ideas as stated by Nguyen Hung et al. (2016). It is worth mentioning that SNs are used not only to spread positive information, but also malicious information. Hence, SNs can be also used for the dissemination of misinformation such as derogatory rumors, disinformation, hate speech, or fake news. For instance, Hosni and Li (2020) shows SNs such as Twitter or Facebook as channels for spreading malicious rumors or misinformation and proposes a method to minimize the spread and influence of these rumors in SNs. These examples motivate the researchers on how to reduce the influence of negative information, resulting in the family of problems known as Influence Minimization Problems Khalil et al. (2013); Luo et al. (2014).

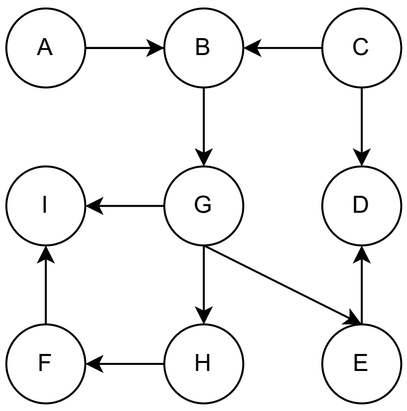
Social Networks are usually represented with a directed graph $G = (V, A)$ where the set of nodes V represents the users, and each relation between two users is modeled as an arc $(u, v) \in A$, with $u, v \in V$, indicating that the user u (departing node) is connected to or can even transmit information to the user v (destination node). In the context of Influence Maximization Problems, information about an initial seed set is also required. Typically, this seed set, denoted as S_0, is limited by a given size l or/and by a given budget b (where it is also required to provide a weight to each node of the graph). In an SN, a user can be active or inactive, depending on whether it has received the information or not. Originally, only nodes in S_0 are active. However, the seed set is responsible for propagating information to the remaining nodes in the network. The propagation process of a given seed set over an SN is evaluated by using an Information Diffusion Model (IDM) Kempe et al. (2003); Lawyer (2015); Lozano-Osorio et al. (2023). Given a seed set, IDM uses a Monte Carlo probabilistic simulation to estimate the number of active nodes. It requires a considerably large number of iterations T (a value of $T = 100$ is widely accepted in the literature) to have a robust evaluation. Given an arc $(u, v) \in A$ with $u, v \in V$, IDM assigns a probability of transmitting information to each arc, denoted as w_{uv}. The way in which this value is assigned depends on the IDM model. Among the most widely used propagation influence methods within the SN can be highlighted: linear threshold model (LTM) Shakarian et al. (2015), trivalency model (TV) Granovetter (1978), weighted cascade model (WCM) Wang et al. (2016), or independent cascade model (ICM) Kempe et al. (2003).

Most of the literature on influence maximization considers the Independent Cascade Model (ICM), where the Monte Carlo simulation generates a random number in the interval [0, 1] for each arc. Then, if this value is less than a small probability (1% is a widely accepted value), the corresponding destination node becomes active; otherwise, it remains inactive. It is worth mentioning that an active node can send information only once to all its inactive neighbors. Then, if there are newly activated nodes, they will be evaluated in subsequent iterations, eventually activating their inactive neighbors. This process stops when no new nodes are activated (e.g., newly activated nodes do not have neighbors, or the random number does not satisfy the threshold constraint). The complexity of that method is $O(I \cdot |V| \cdot |A|)$ where I is the numbers of iterations. Independently of the influence diffusion model considered (LTM, TV, WCM, ICM, among others) it is a highly computationally demanding method mainly for large-scale networks, being a less scalable method than other deterministic alternatives Ravelo and Meneses (2021). In order to deal with this drawback, Liu et al. (2012) proposed Exponent Supervised Monte Carlo Estimation (ESMCE), which efficiently estimates the influence spread by randomly sampling only a portion of the neighbors of each incumbent node. Although ESMCE can considerably accelerate the process, it has an obvious impact on its accuracy. Many other algorithms and heuristics have also been proposed to overcome this efficiency issue, such as Chen et al. (2010), Liu et al. (2014), Wang et al. (2023), Wang and Liu (2023) and Arepalli et al. (2019).

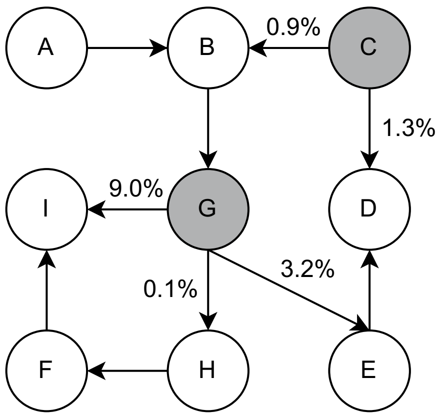
Figure 1 a shows an SN with 9 nodes and 10 arcs. Let us assume that the size of the seed set is $l = 2$. In this example, it has been considered that the seed set is composed of nodes G and C. Notice that any pair of nodes could be considered as a seed set. Figure 1 b illustrates how the seed set influences its neighbors, being the set of candidate nodes to be activated (e.g., {B, D, E, H, I}). Additionally, a random probability is represented by a number close to each arc. Note that it is only required to generate probabilities for adjacent nodes to those in the seed set. As was aforementioned, a threshold of 1% is usually considered. Therefore, nodes B and H are activated (highlighted with a dark gray background). Figure 1 c depicts the next iteration, where the candidates are now the neighbors of the active nodes detected in the previous iteration. Specifically, the candidate to be activated is F, where the nodes already active are no longer considered. Again, random probabilities are represented close to each arc. In this example, only node F is activated. Finally, in Fig. 1 d, the set of candidates to be activated is {I}. However, random probabilities are larger than the threshold. Therefore, no new nodes are influenced, ending the Monte Carlo simulation, the total number of activated nodes being equal to 5. It is worth mentioning that there are some nodes that cannot be influenced by any other node. In particular, every node with an in-degree equal to zero cannot be influenced (see, for instance, node A in Fig. 1 a).

It is important to remark that only one iteration (out of T) has been represented in the Monte Carlo simulation. The final solution is computed by averaging the active nodes across the number of simulations considered. Without loss of generality, has been considered a maximization approach (the larger the resulting averaged value, the better the solution is). This optimization problem has been commonly

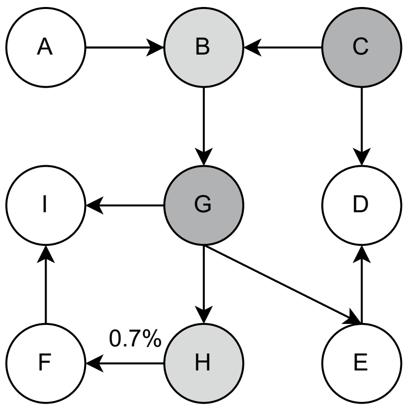
Fig. 1 Monte Carlo simulation example considering $l = 2$



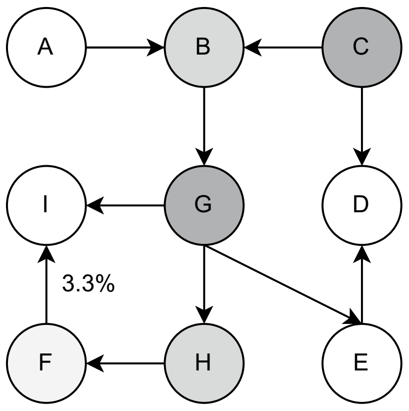
(a) Social Network with 9 nodes and 10 arcs



(b) Seed set {`C`,`G`} and activation probabilities.



(c) New activated nodes {`B`,`H`} and activation probabilities.



(d) Node `F` is activated and activation probability to `I`.

referred to in the related literature as the Social Network Influence Maximization Problem (SNIMP) Gong et al. (2016); Lozano-Osorio et al. (2021). Notice that there are some approaches that consider a minimization approach (see Sect. 2 for further details).

Before defining SNIMP more formally, let IDM_i be the i-th simulation of the IDM model (with $1 \leq i \leq T$). Then, for simplicity, it was assumed that IDM_i is a function that receives three input parameters: the social network G, the seed set S_0, and the activation probability ρ; and returns a set S_1^i (with $S_0 \subseteq S_1^i$) that refers to the set of influenced nodes. More formally:

$$S_1^i \leftarrow IDM_i(G, S_0, \rho)$$

As was aforementioned IDM is an stochastic process. Therefore, the resulting set of the i-th simulation (S_1^i) may differ from the one obtained in the j-th simulation (S_1^j).

The objective function of SNIMP is computed as the expected statistic value across the T Monte Carlo simulations. Then, the optimization problem consist in finding the initial seed set S_0 that maximizes the objective function. In mathematical terms:

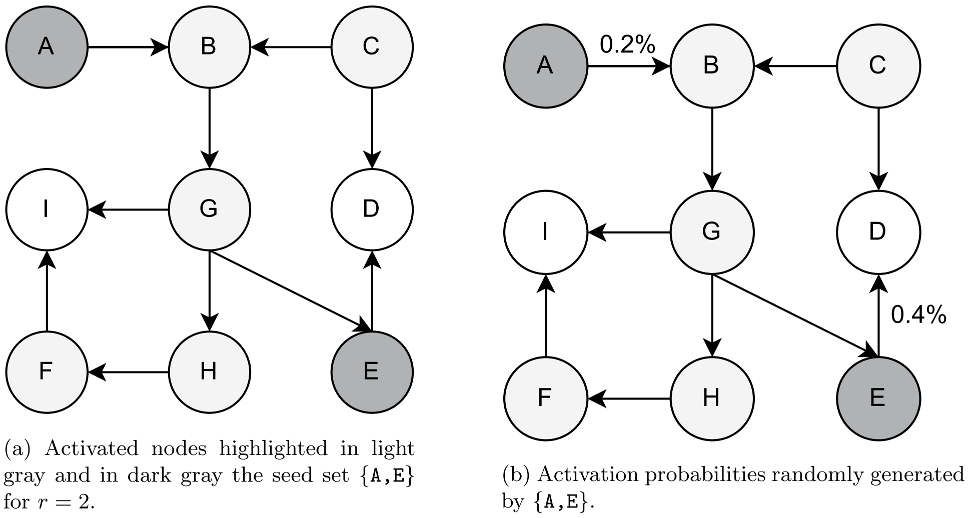
$$S_1^{\star} \leftarrow \arg\max{}_{S_0 \in \mathbb{S}} \mathbb{E}[|\bigcup_{i=1}^{T} IDM_i(G, S_0, \rho)|]$$

being $\mathbb{S}$ the set of all combinations of l elements taken from the set of nodes V.

The Multi Round Influence Maximization (MRIM) problem can be considered as a generalization of the SNIMP, where considering R different rounds. It is equivalent to the well-studied Social Network Influence Maximization Problem (SNIMP) when considering $R = 1$, so both problems are $\mathcal{NP}$-hard as stated in Kempe et al. (2015).

MRIM originally emerged in the context of viral marketing, where advertisers conduct multiple rounds of viral marketing to promote one product Hinz et al. (2011). In the related literature, this problem has been approached by considering two perspectives. On the one hand, the non-adaptive MRIM (NMRIM) variant, where the advertiser establishes seed sets for all rounds at the very beginning.

Fig. 2 Previous example when more rounds are considered



(a) Activated nodes highlighted in light gray and in dark gray the seed set {A,E} for $r = 2$.

(b) Activation probabilities randomly generated by {A,E}.

On the other hand, the adaptive MRIM (AMRIM) variant, where the advertiser can select seed sets adaptively (where each round is partially related to the next one) based on the propagation results in the previous rounds. In this paper, the second approach is the primary focus, since NMRIM can be directly solved by applying R times any SNIMP algorithm.

From a mathematical perspective, the main difference between NMRIM and AMRIM is the set of nodes where the initial seed set can be selected. In the former, it is possible to select from any set of $\mathbb{S}$ (which contains all combinations of l nodes taken from V) in each round. This fact illustrates the non-adaptive nature of the NMRIM since information about activated nodes is not used from one round to the next one. In the latter problem, $\mathbb{S}$ is modified from one round to the next one by incorporating information (activated nodes) from previous rounds. More precisely, given the round r (with $1 \leq r \leq R$), let us define the set of "eligible" nodes V^r as follows:

$$V^r \leftarrow V^{r-1} \setminus S_1^{r-1}$$

where V^0 is the set of nodes of the original graph and $S_1^0 = \emptyset$. Notice that in each round, activated nodes are discarded since AMRIM looks for the maximization of influenced users and those users where already activated in previous rounds.

As described in the related literature Sun et al. (2018a), the optimal solution of the AMRIM problem is represented as a vector of r components, denoted as $\boldsymbol{S}^\star = [S_1^{\star,1}, S_1^{\star,2}, \dots, S_1^{\star,R}]$ where each component $S_1^{\star,r}$ (with $1 \leq r \leq R$) is computed as:

$$S_1^{\star,r} \leftarrow \arg\max_{S_0 \in \mathbb{S}^r} \mathbb{E}[|\bigcup_{i=1}^{T} IDM(G, S_0^i, \rho)|]$$

being $\mathbb{S}^r$ the set of all combinations of l elements taken from the set of nodes V^r.

Departing from the example depicted in Fig. 1, the evaluation of a solution for the AMRIM is illustrated by considering $R = 2$ and $l = 2$. Specifically, the first round ($r = 1$) was depicted in Fig. 1d where, with $S_0^1 = \mathtt{C,G}$, $S_1^1 = \mathtt{C,G,B,H,F}$ is obtained, resulting in the objective function value of:

$$\mathbb{E}[|\bigcup_{i=1}^{1} IDM(G, \{\mathtt{C,G}\}, 0.1)|] = \mathbb{E}[|\{\mathtt{C,G,B,H,F}\}|] = 5.$$

In Fig. 2, the activated nodes of the first round are highlighted with light gray. Additionally, selected nodes being included in the next seed set are highlighted with dark grey (i.e., $S_0^2 = \{\mathtt{A,E}\}$). Notice that these nodes are selected from the set $V^2 = V \setminus S_1^1 = \{\mathtt{A,D,E,I}\}$. Indeed, the set of possible seed set constructed by considering all combinations or $l = 2$ nodes taken from V^2 is $\mathbb{S}^2 = \{(\mathtt{A,D}), (\mathtt{A,E}), (\mathtt{A,I}), (\mathtt{D,E}), (\mathtt{D,I}), (\mathtt{E,I})\}$. Notice that this set can be further processed by excluding those nodes without outgoing arcs, since they cannot activate any neighbor.

Figure 2 b includes the information about probabilities, which are the numbers close to the arcs $(\mathtt{A,B})$ and $(\mathtt{E,D})$. Let us consider that the result of the corresponding Monte Carlo simulation activates nodes $\{\mathtt{D,B}\}$. It is worth mentioning that node B was activated in the previous round. Therefore, it will not be activated again. In this example, the Monte Carlo simulation stops since node D has no adjacents. Then, the result of the objective function for $r = 2$ is calculated as follows:

$$\mathbb{E}[|\bigcup_{i=1}^{2} IDM(G, S_0^i, 0.1)|]$$
$$= \mathbb{E}[|IDM(G, \{\mathtt{G,C}\}, 0.1) \cup IDM(G, \{\mathtt{A,E}\}, 0.1)|] = 8$$

Finally, considering the two rounds of this example, the solution vector is $\boldsymbol{S}= [5, 8]$.

This paper presents a novel metaheuristic approach to deal with AMRIM, which allows us to find high-quality solutions in a reasonable computing time. Our main goal is to design an efficient algorithm to find the most influential users in an SN, considering the ICM as the IDM for evaluating solutions. An algorithm based on the Variable Neighborhood Search (VNS) methodology is presented, characterized by its efficiency when designing solutions for $\mathcal{NP}$-hard combinatorial optimization problems. The proposed procedure is validated over a set of widely used real-world instances in the context of social influence maximization and compared against the state-of-the-art method based on a totally greedy approach AdaIMM Sun et al. (2018b). The results obtained show the efficiency and efficacy of the proposed methodology.

The main contributions of this work are the following.

- A metaheuristic algorithm based on the Variable Neighborhood Search methodology applied to SN problems is proposed.
- The effect of each component of the proposed algorithm is carefully analyzed to verify the contribution of each stage to the final results.
- A constructive procedure to generate initial solutions based on independent probability based on events is presented.
- An intelligent local search method based on community detection is proposed.
- The dataset is increased with more complex and challenging instances, where optimal values are unknown, and exact algorithms are unable to find a solution.
- Competitive testing is performed with the VNS algorithm and state-of-the-art algorithms.
- The source code of the proposed algorithm and instances is publicly available to ease further comparisons, as well as the complete results.[1]

The remaining document is organized as follows. Section 2 defines the literature of AMRIM, describing the most successful approaches. Section 3 describes the proposed algorithm and the strategies used to solve it. Section 4 presents the computational results including a competitive testing with the state-of-the-art algorithm for AMRIM. Finally, the conclusions and future research are discussed in Sect. 5.

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

2 Literature review

The problem of selecting the most influential users in SNs was originally formulated in Richardson and Domingos (2002). Kempe et al. (2003) were the first to solve the original problem, SNIMP, formulating it as a discrete optimization problem. The authors 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 that are always within a factor of at least 63% of the optimal value under the IDMs described in Sect. 1. Then, Kempe et al. (2015) stated that SNIMP is $\mathcal{NP}$-hard.

As a consequence of the computational effort required to evaluate the ICM, Kempe et al. (2003) also proposed several greedy heuristics based on metrics from the SN analysis, 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 but losing efficacy. When the metric considered is the degree of the node, the algorithm is called *high-degree heuristic*.

Later, some greedy algorithms were introduced. Leskovec et al. (2007) proposed the Cost-Effective Lazy Forward (*CELF*) selection where they reduce the computational time using the submodularity property. Then, Goyal et al. (2011) proposed *CELF++* with the aim of improving the efficiency of the original *CELF* resulting in a method 35–55% faster than the original *CELF*.

It is worth mentioning the increase of open source tools such as Social Network Visualizer (SocNetV), which ease the analysis of social networks by the visualization of connection structures and patterns Wu et al. (2020).

The interest in influence maximization problems in the last years has been continuously increasing, resulting in a wide variety of new problems derived from the original SNIMP and methods for solving them. We refer the reader to Aghaee and Kianian (2020); Aghaee et al. (2021); Bouyer et al. (2023); Jaouadi and Ben Romdhane (2024) for recent success research works on this family of problems. Recently, Jaouadi et al Jaouadi and Ben Romdhane (2024) shows that metaheuristics are still scarce in the area and machine learning approaches emerges, being Li et al. (2023) a recent study in machine learning approaches in SNIM.

Golovin and Krause (2010) were the first to study influence maximization under the adaptive model, assuming that the l seed nodes are chosen sequentially, such that the selection of the $(i + 1)$-th node is performed after the influence of the first i nodes has been observed. The authors proposed a simple greedy algorithm for adaptive influence maximization that returns a seed set S whose influence is, at least, $1 - 1/e$ of the optimum value in the case where

Table 1 Algorithms for solving the MRIM in the literature. The notation of the complexity analysis as presented by the original authors are: k is the size of the seed set added per round, m the IDM complexity, T is the number of rounds, n is the number of users in the SN, and ϵ is the set of all the possible items (S_t, t)

Algorithm	Method	Instances	IDM	Complexity
SG	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O(n \cdot m)$
SG-R	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O(n \cdot m)$
CR-Greedy	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O\left(k^3 \ell T^4 n^2 m \log(n)/\epsilon^2\right)$
WR-Greedy	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O\left(k^3 \ell T n^2 m \log(nT)/\epsilon^2\right)$
AdaGreedy	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O\left(k^3 \ell T n^2 m \log(nT)/\epsilon^2\right)$
AdaIMM	Greedy	Barbieri et al. (2012); Chen (2008)	ICM	$O(T(k+\ell)(n+m)\log(nT)/\epsilon^2)$

only one seed is selected in each seed set. However, the algorithm requires knowing the exact expected influence of every node, which is impractical, since the computation of the expected spread is $\mathcal{NP}$-hard in general, as stated in Wang et al. (2012). Vaswani and Lakshmanan (2016) extend Golovin and Krause (2010) model by improving the errors in the estimation of expected spreads. Their method returns an $\left(1 - e^{-(1-1/e)^2/\eta}\right)$-approximation under this setting, where η is a certain number greater than 1.

Recent works such as Huang et al. (2020) study several efficient approximation algorithms in the context of non-adaptive Multi Round Influence Maximization, proposing as future work tackling the adaptive variant. Chen et al. (2022) conclude in a similar way in their research of non-adaptive and adaptive problems, where the adaptive variants for ICM and LTM result in challenging open problems.

To mitigate the drawback of the requirement of knowing the exact expected influence of every node in the adaptive version, Sun et al. (2018b) and the extension Sun et al. (2018a) provide a further analysis about Multi Round Influence Maximization in an adaptive and non-adaptive way, comparing it with several methods in the state-of-the-art. Their proposal, in the worst-case, guarantees an approximation of $1 - e^{(1-1/e)(\epsilon-1)}$ with high probability, where $\epsilon \in (0, 1)$ is a user-specified parameter. The outperforming method is a greedy method called AdaIMM, which is considered the state of the art for this problem, so it will be included in the competitive testing in Sect. 4. Table 1 shows a summary of the main features of the most relevant works in the context of MRIM.

All the methods directly focused on solving the MRIM, included in Table 1, were presented in 2018. It is worth mentioning that all the algorithms used the Independent Cascade Model as IDM and that the instances' column provides the reference for when each instance was proposed. We refer the reader to Table 2 for more detailed information on these instances: Flixster and NetHEPT. SG and SG-R are similar methods: both select $T \times k$ seed nodes using a single-round greedy algorithm. The main difference relies on that SG-R uses the same seeds for all rounds, while SG provides different seeds for each round. CR-Greedy iteratively selects the user with the maximum marginal influence spread without replacement. If the budget for a specific round is exhausted, the remaining nodes in are removed from the candidate list. Since the candidate list includes nodes assigned to different rounds, CR-Greedy can select nodes across multiple rounds. The idea of WR-Greedy is that seed nodes are selected round by round. More specifically, it greedily selects seed nodes for the first round, and then continues selecting seed nodes for the next round. AdaGreedy takes the set of already activated nodes as feedback from the previous rounds and aims to find the seed set that maximizes the expected marginal gain. To achieve this, a Monte Carlo greedy approximation (MC-Greedy) algorithm is used to find an approximate solution. Finally, AdaIMM consists of two phases: *Sampling* and *Node Selection*. In the Sampling phase, the algorithm iteratively generates random sets of nodes and accumulates them until a stopping condition is met. In the *Node Selection* phase, the standard greedy algorithm for maximum coverage is applied to find a node set that maximizes coverage over the random sets previously selected. The greedy algorithm for maximizing coverage is inspired by the idea originally proposed in Tang et al. (2015).

Finally, we refer the reader to some recent surveys Banerjee et al. (2020); Aghaee et al. (2021), where it is shown that Sun et al. (2018a) is considered the state-of-the-art proposal. Additionally, those studies highlight that metaheuristics have been mainly ignored in the context of SNs problems.

3 Variable neighborhood search

Variable Neighborhood Search (VNS) methodology was originally proposed as a simple metaheuristic Mladenović and Hansen (1997) for solving hard optimization problems. The main contribution of this algorithmic approach is to consider several neighborhoods during the search and to perform systematic changes in the neighborhood structures. Although it was originally presented as a simple metaheuristic, the success of the methodology has lead to the proposal of several new variants: Reduced VNS (RVNS), Variable Neighborhood

Descend (VND), General VNS (GVNS), Variable Neighborhood Decomposition Search (VNDS), Skewed VNS (SVNS) or Variable Formulation Search (VFS), among others. See Hansen et al. (2010, 2009); Lozano-Osorio et al. (2020) for a detailed analysis of each variant.

In this work, the Basic Variable Neighborhood Search (BVNS) is used, one of the original variants. BVNS is known for its simplicity and efficiency when dealing with hard optimization problems. The algorithm consists of three main parts: a shaking phase, an intensification method, and a neighborhood change step. Algorithm 1 shows the pseudocode of BVNS.

Algorithm 1 *BVNS*(k_{max},R, l)

```
1:  $S \leftarrow \emptyset$
2:  for $r \in 1 \dots R$ do
3:      $S \leftarrow CreateRound(S, l)$
4:      $S \leftarrow LocalSearch(S)$
5:      $k \leftarrow 1$
6:      while $k \leq k_{\max}$ do
7:          $S' \leftarrow Shake(S, k)$
8:          $S'' \leftarrow LocalSearch(S')$
9:          if $f(S'') > f(S)$ then
10:             $S \leftarrow S''$
11:             $k \leftarrow 1$
12:         else
13:             $k \leftarrow k + 1$
14:         end if
15:     end while
16: end for
17: return $S$
```

The algorithm receives three input parameters: the largest neighborhood to be explored, k_{max}; the number of rounds used, R; and, finally, the size of the seed set per round l, resulting in the complete seed set of size $l * R$. The algorithm starts by creating the seed set S where it contains the selected nodes (step 1). Steps 2-16 represent the number of rounds used in the AMRIM problem, and in step 3, an initial solution S is generated considering the constructive procedure presented in Sect. 3.1. The solution is then locally improved with the local search method described in Sect. 3.2 (step 4). Starting from the first predefined neighborhood (step 5), BVNS iterates until it reaches the maximum considered neighborhood k_{max} (steps 6-15). For each iteration, the incumbent solution is perturbed by the shake method (step 7). This method is designed to escape from local optima by randomly exchanging the position of k nodes, generating a solution S' in the neighborhood under exploration. The local search method (step 8) is then responsible for finding a local optimum S'' in the current neighborhood with respect to the perturbed solution S'. Finally, the neighborhood change method selects the next neighborhood to be explored (steps 9-14). In particular, if S'' outperforms S in terms of the objective function value, then it is updated (step 10), and the search starts again from the first neighborhood (step 11). Otherwise, the search continues in the next neighborhood (step 13). The current round stops when reaching the largest neighborhood considered k_{max}, then starts the new round using the best seed set S. Finally, when the number of rounds ends, the best seed set found during the search is in S and returns this value (step 17).

To further illustrate the operational process of the BVNS algorithm, Fig. 3 provide a detailed flowchart. Specifically, it visually represents the step-by-step progression of the algorithm, highlighting key stages such as constructive, local search and shake procedure.

BVNS starts from an initial solution that can be generated either at random or with a more elaborate procedure (see Sect. 3.1). Then, the solution is locally improved to start the search from a local optimum (see Sect. 3.2). Afterwards, the method starts by exploring the selected neighborhoods $\{N_1, \dots, N_{k_{max}}\}$. In each iteration, the shake procedure (see Sect. 3.3) perturbs the incumbent solution in the current neighborhood and then finds a local optimum in this new neighborhood (see Sect. 3.2). Finally, if an improvement has been found, the search starts again with the first

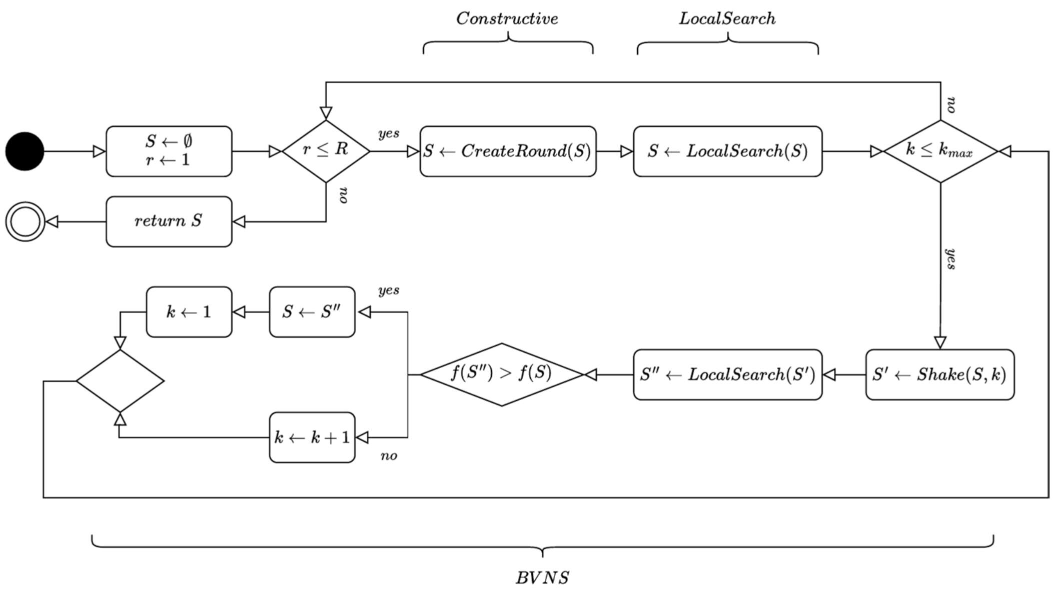


Fig. 3 BVNS flowchart

neighborhood. Otherwise, it continues exploring the next available neighborhood.

3.1 Constructive algorithm

As stated in Mladenović and Hansen (1997), VNS designs require from an initial solution to start the search, excluding the way in which that initial solution is constructed from the methodology. In fact, some authors have even considered starting the search from a feasible random solution Herrán et al. (2019); Yuste et al. (2023). However, recent studies Pérez-Peló et al. (2021); Lozano-Osorio et al. (2022) have experimentally shown that providing a high-quality starting point fosters the algorithm to focus the search on more promising regions of the search space. In order to validate this hypothesis, in this research both has been considered: random and greedy approaches.

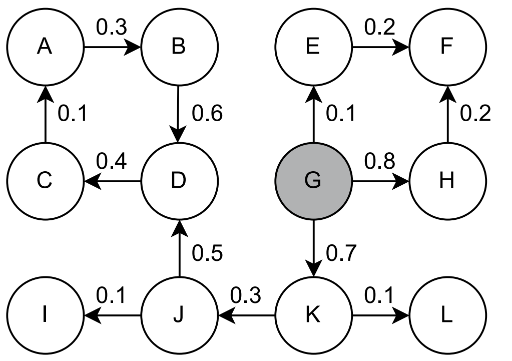
The random approach, denoted as $\boldsymbol{C_{RND}}$, for the AMRIM problem focuses only on constructing a feasible solution, which is represented as a vector $\boldsymbol{S}$ with R components, then the complexity is $O(R)$. Specifically, for the first round ($r = 1$), it randomly selects l nodes from the set of nodes V. As was aforementioned, this set is called S^r_0. After that, a Monte Carlo simulation is conducted, obtaining the set of activated nodes S^r_1, becoming into the first component of the vector $\boldsymbol{S}$. For the second round ($r = 2$), the set of eligible nodes is updated as $V_1 = V \setminus S^1_1$ (see Sect. 1 for further details). Then, l new elements are selected from V_1 and conducted again a new Monte Carlo simulation, becoming into the second component of $\boldsymbol{S}$. This process is maintained until $r = R$, returning a feasible solution for the AMRIM problem.

The greedy approach, denoted as $\boldsymbol{C_{GR}}$, is inspired by a recent approach described in Bouyer et al. (2023), where the authors proposed a probabilistic approach for a different optimization problem. Specifically, this method considers the node activation probability as a heuristic value, with the aim of first including those nodes with the minimum probability of being activated in the incumbent solution. In particular, given a round r, a partial solution in that round S^r_1, and a node $v \notin S^r_1$, in this research the function $g_{GR}(v)$ has been introduced, it estimates the probability of v of being activated when considering those nodes in S^r_1. This method starts by considering the node with the largest degree; it is selected as the starting seed node (ties are broken randomly). Then, the probability of activation w_{uv} associated to each arc $(u, v) \in A$ between each pair of nodes $u, v \in V$ is computed by using the corresponding IDM. After that, the probability of activation of each node of the SN is calculated as the conditional probability of independent events.

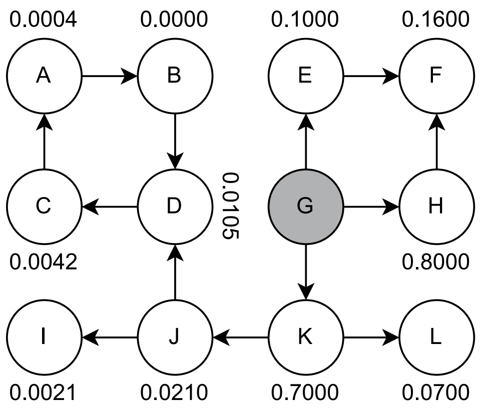
Before defining this probability more formally, let us first define a path ϕ_{uv} in graph G, where u is an activated node ($u \in S^r_1$) and v is a non-activated node ($v \notin S^r_1$). For the sake of brevity, ϕ_{uv} is represented as a set of nodes $\{u, \dots, v\}$ whose size vary from 2 (both nodes are adjacent) to $|V| - 2$ (all nodes of the graph belong to the path). Finally, the i-th element in the path is identified as $\phi_{uv}(i)$ (with $1 \le i \le |\phi_{uv}|$).

Given a round r, a partial solution S^r_1, and an active node $u \in S^r_1$, the activation probability of a node $v \notin S^r_1$ is computed as:

$$P(u \cap v) \leftarrow \prod_{i=1}^{|\phi_{uv}|-1} w_{\phi_{uv}(i), \phi_{uv}(i+1)}$$



(a) Social Network with 12 nodes and 12 relations.



(b) Probability of node activation when node G is activated.

Fig. 4 Example of SN and the node activation probability

Taking into account the intention of including in the solution those nodes with a low probability of activation and, additionally, to force exploration of different regions of the SN, $g_{GR}(v)$ is defined as:

$$g_{GR}(v) \leftarrow \min_{u \in S_1^r} P(u \cap v)$$

Therefore, the candidate to be incorporated in the solution under construction is selected as the one with the minimum probability of being activated by the already selected nodes. In mathematical terms:

$$v^\star \leftarrow \arg\min_{v \notin S_1^r} g_{GR}(v)$$

Let us illustrate how this value can be evaluated with an example. Figure 4 a represents an SN with 12 nodes and 13 arcs. Node G is selected as the first node in the seed set (highlighted with a dark gray background), chosen at random from the set of nodes with the largest degree; i.e., {G, K, J}. Once at least one activated node is available, a Monte Carlo simulation can be executed to determine the activation probability (see the real numbers close to each arc of Fig. 4 a).

In Fig. 4 b, the actual probability of activating each node is shown by considering the above-mentioned equation. For example, in order to determine the probability of activation of node E, the corresponding path is first identified, $\phi_{GD} = \{G, K, J, D\}$, and then $P(G \cap D) = w_{GK} \cdot w_{KJ} = 0.0105$. Notice that the number of paths between pairs of nodes exponentially grows with the size of the SN. This situation is overcome by constructing only one path between each pair of nodes. In particular, given a node in the seed set, a Breadth-First Search (BFS) is conducted, where the nodes in each level are explored according to the probability of each arc. BFS returns an spanning tree, where only one path exists between the root and the rest of the nodes of the tree. For example, considering the SN depicted in Fig. 4 a, BFS starts by considering as root the node G; then, nodes {E, H, J} are explored. Considering the probabilities of arcs $P(G \cap E) = w_{GE} = 0.1000$, $P(G \cap H) = w_{GH} = 0.8000$ and $P(G \cap K) = w_{GK} = 0.7000$, the selected node to continue the search is H. In order to increase the efficiency of BFS, a priority queue is used, where elements are sorted according to the probability.

Specific values of probabilities assigned to each node are depicted in Fig. 4 b. Then, the selected node is the one with the lowest probability (i.e., node B). The next iteration of the constructive method proceeds in a similar way: first, execute a new Monte Carlo simulation considering that nodes {B, G} are in the seed set, obtaining the probabilities of each arc; second, construct a BFS from each node in the seed set, computing the probability of each path; and third, assign the largest computed probability to each node in the SN. The node selected to be incorporated into the seed set is the one with the smallest probability. Ties are broken by first considering the node with the largest out-degree. The complexity of this method is $O(|V| + |A|)$, since it is equivalent to a breadth-first search.

The impact and influence of each constructive procedure on the generated solutions will be analyzed in more detail in Sect. 4.1.

3.2 Local search

The improvement phase in VNS is devoted to reach a local (ideally global) optimum with respect to the neighborhood under exploration. Designing a local search for influence

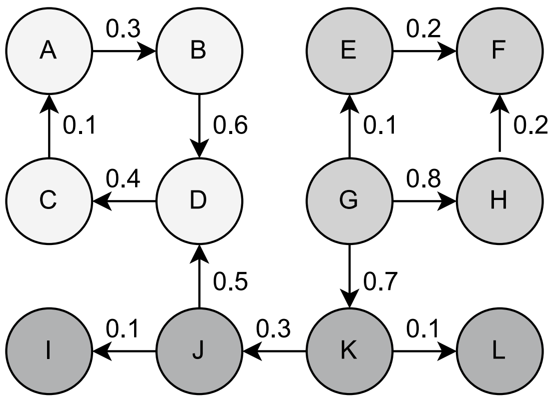


Fig. 5 Communities according to Louvain method

maximization problems usually results in rather complex algorithms, which are sometimes impractical due to the computational requirements.

The proposed local search is related to a well-studied common feature in SNs, the community structure. Girvan and Newman (2002) define that the networks with this property have the capacity to be divided into groups in such a way that the connections among users in the same group are dense while connections among users in different communities are sparse.

As shown in recent related literature Pérez-Peló et al. (2019); Chunaev (2020), there exists a wide variety of community detection algorithms. Blondel et al. (2008) proposed Louvain algorithm, a method divided into two phases: the first one follows an agglomerative approach, considering that each node is initially a community by itself in the network. It then joins communities optimizing the modularity metric, and it stops when a local maximum of modularity is reached.

Modularity Newman and Girvan (2004) is a metric designed to evaluate the probability of the existence of an edge connecting two nodes in the graph in another random graph. The higher the modularity, the better the community detection is. The modularity of a community c over a community detection CD for a graph G is defined as follows:

$$Md(c, CD, G) = (e_{jj} - a_j^2)$$
$$e_{cc} = \frac{|\{(v,u) \in E \ : \ CD(v) = CD(u) = c\}|}{|E|}$$
$$a_k = \frac{|\{(v,u) \in E \ : \ CD(v) = c\}|}{|E|}$$

where c_{max} is the number of communities in the solution, e_{cc} is the percentage of intra-community edges (with respect to the whole set of edges) in the community c, and a_c is the percentage of edges with at least one endpoint in c. Then, the modularity of the complete community detection CD is formally defined as:

$$Md(CD, G) = \sum_{c=1}^{c_{max}} Md(c, CD, G)$$

In the second phase, the algorithm builds a new network, merging the nodes in the same community into a single node, and performs again the first phase until no improvement in modularity is found. The authors report that this method has a complexity of $O(|V|)$.

Figure 5 shows the different communities selected by the Louvain method: the first one conformed by nodes `A, B, C`, and `D`; the second one conformed with `E, F, G`, and `H`; and the last one conformed with nodes `I, J, K`, and `L`. Notice that the number of edges connecting different communities is extremely small (a maximum of 1), while the number of edges inside each community is larger, which indicates that the community detection method has identified the community structure of the graph.

The neighborhood of a solution S_0^i is defined as the set of solutions that can be reached by performing a single move over S_0^i. The move considered is a swap move $Swap(S_0^i, u, v)$ where node u is removed from the seed set, being replaced by v, with $u \in S_0^i$ and $v \notin S_0^i$. This swap move is formally defined as:

$$Swap(S_0^i, u, v) = (S_0^i \setminus \{u\}) \cup \{v\}$$

Thus, the neighborhood $N_s(S_0^i)$ of a given solution S_0^i consists of the set of solutions that can be reached from S_0^i by performing a single swap move. More formally,

$$N_s(S_0^i) = \{Swap(S_0^i, u, v) \quad \forall u \in S_0^i, \forall v \in V \setminus S_0^i\}$$

The size of the resulting neighborhood, $l \cdot (N - l)$, makes the complete exploration of the neighborhood not suitable for AMRIM, even considering an efficient implementation of the IDM.

To determine the reduced neighborhood that will be explored, the Louvain community detection algorithm is considered. Specifically, the method generates the communities for a given solution with the aim of analyzing the number of nodes activated in each community. This intelligent neighborhood exploration strategy is denoted as $\boldsymbol{LS_{cd}}$, and is designed to reduce the number of solutions explored within each neighborhood, without drastically deteriorating the quality of the obtained solutions. This reduction in the size of the search space is performed by exploring just a small fraction of the nodes belonging to each community for the swap move. For each community, only those nodes which are not yet in the solution with the largest degree are considered in the swap move to be included in S_0^i. The communities are explored in descending order with respect to the number of non-active nodes,

Table 2 Features of the instances used in this work

Instance	Nodes	Arcs	Min(deg)	Max(deg)	Avg(deg)	Diameter	Clust. Coeff.	Refs.
WikiVote	7115	103689	1	1065	28.32	7	0.1409	Lozano-Osorio et al. (2021); Bucur and Iacca (2016)
NetHEPT	15233	32235	1	64	4.12	22	0.4984	Sun et al. (2018b); Chen (2008)
ca-AstroPh*	18772	198110	1	504	21.11	14	0.6306	Lozano-Osorio et al. (2021); Liu et al. (2019)
ca-CondMat*	23133	93497	1	281	8.08	14	0.6334	Liu et al. (2019); Lawyer (2015)
cit-HepPh	34546	421578	1	846	24.37	12	0.2848	Liu et al. (2019); Lawyer (2015)
email-Enron*	36692	183831	1	1383	10.02	11	0.4970	Liu et al. (2019); Malliaros et al. (2016)
p2p-Gnutella31	62586	147892	1	95	4.73	11	0.0055	Liu et al. (2019); Lawyer (2015)
Flixster	95969	484865	1	428	10.10	14	0.1086	Sun et al. (2018b); Barbieri et al. (2012)

i.e., the community with the largest number of non-active nodes is the first to be explored. In terms of complexity, it is difficult to analyze the exact number of iterations that a local search perform, since it is usually iterating until no improvement is found. Therefore, the complexity reported is the one for a single iteration. First of all, it is necessary to evaluate the complexity of the Monte Carlo simulation, which is $O(I \cdot |V| \cdot |A|)$ (where I is the number of Monte Carlo simulations). Then, iterating over the neighborhood has a complexity of $O(|CD| \cdot |V|)$. Then, the complete complexity of the method is $O(I \cdot |V|^2 \cdot |A| \cdot |CD|)$. Notice that, the surrogated local search has similar complexity but reducing the number of iterations I, thus reducing the final computing time.

In order to compare our proposal, in Sect. 4.1 the local search proposed in this research is compared against the local search presented in Lozano-Osorio et al. (2021) denoted as $\boldsymbol{LS_{sn}}$. The main drawback of this proposal is the computational cost of the objective function evaluation, which is the most demanding part of the proposed algorithm. For this reason, the authors limit the number of required simulations and nodes required to be explored in the neighborhood, leading to a more efficient but less accurate procedure.

3.3 Shake

The perturbation mechanism in VNS is usually called the shake procedure. The goal of this method is to diversify the search by randomly generating a neighbor solution of the incumbent one, which may eventually lead to further regions of the search space. We propose a method that modifies the structure of the solution according to a parameter k. Its value ranges from 1 to k_{max}, which is an input parameter of the complete procedure. As it is customary in VNS, it is not recommended to consider large values for $k_{\max}$ since it may result in a solution which is equivalent to construct a new one. This method has a complexity of $O(k_{\max})$, since no Monte Carlo evaluation is performed.

The proposed shake method performs k swap moves to the incumbent solution. As it is customary in the BVNS methodology, these elements are selected at random. Recent works have studied more advanced shake techniques that balance diversification and intensification of the search. This strategy has been referred to as intensified shake (see Duarte et al. (2014) and Sánchez-Oro et al. (2014) for further details).

However, in the context of AMRIM, both the constructive and local search focus completely on intensification, so it is interesting that the shake procedure focuses on diversification to achieve a balance between them. Therefore, in this research, only the traditional random shake procedure has been considered.

The final complexity of a single BVNS iteration is evaluated as the maximum between the shake complexity, which is $O(k_{\max})$, and the local search method complexity, which is $O(I \cdot |V|^2 \cdot |A| \cdot |CD|)$, resulting in the complexity of the local search method, i.e., $O(I \cdot |V|^2 \cdot |A| \cdot |CD|)$.

4 Computational results

This section describes the computational experiments designed to evaluate the performance of the proposed algorithms and to analyze the results obtained. All experiments have been performed on an AMD EPYC 7282 16-core virtual CPU with 32GB of RAM. The operating system used was Ubuntu 20.04.2 64 bit LTS, and all algorithms were implemented in Java 17 using the *Metaheuristic Optimization frameweoRK* (MORK) 13 Martín et al. (2022). The testbed of instances used in this work is the same set considered in the previous work (NetHEPT and Flixster), also including new competitive instances that have been extensively used in the context of influence maximization problems. Table 2 summarizes the instances considered in this work.

In particular, the table shows the features per instance, including: the name of the dataset; number of nodes; number of arcs; minimum, maximum, and average node degree; maximum distance between the pair of vertices denoted as

Table 3 Comparison between a random, IDM based and greedy construction

Algorithm	Avg.	Dev.	Time	#B
$\boldsymbol{C_{RND}}$	2731.45	45.13	**0.21**	0
$\boldsymbol{C_{IDM}}$	2582.34	49.62	0.29	1
$\boldsymbol{C_{GR}}$	**4255.66**	**0.01**	1.49	**9**

Best results are highlighted with bold font

diameter; and the clustering coefficient, which is a measure of the degree to which nodes in a graph tend to cluster together, denoted as Clust. Coeff. and related Social Influence Maximization works where the instances are used. Notice that, the instances marked with and "*" are undirected and the remaining are directed.

First of all, it is worth mentioning that the IDM used is ICM, which is directly derived from the state-of-the-art work. In the ICM, it is important to indicate the values for its parameters and the corresponding Monte Carlo simulation. In all the experiments, as stated in Sect. 3.1, 100 simulations of the ICM are performed with a probability of influence of 0.01. These parameter values are the most extended in the related literature. The number of rounds R, as stated in the state of the art, is set to $R = 5$, and the number of seed nodes selected in each round l is set to 10 as stated in Sun et al. (2018b).

The experiments are divided into two parts: preliminary and final experimentation. The former (Sect. 4.1) refers to the 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, which is AdaIMM Sun et al. (2018b), and a recent method **IMP** $+\boldsymbol{LS}_{sn}$ based on supervised learning approach Lozano-Osorio et al. (2024), which states that the good candidates nodes are those with high in-degree and average probability of neighboring higher than 2 (with a complexity of $O(|V|)$, additionally has been included to improve the solution the previously mentioned local search $\boldsymbol{LS}_{sn}$Lozano-Osorio et al. (2020), with a complexity of $O(l \cdot \delta \cdot I \cdot |V| \cdot |A|)$. It is worth mentioning that the method has been selected since a recent study Jaouadi and Ben Romdhane (2024) concluded that machine learning approaches emerge in the area.

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

Table 4 Comparison between both local search strategies

Algorithm	Avg.	Dev.	Time	#B
$C_{GR} + \boldsymbol{LS}_{sn}$	4257.68	0.23	**2.73**	3
$C_{GR} + \boldsymbol{LS}_{cd}$	**4264.23**	**0.07**	3.75	**7**

Best results are highlighted with bold font

4.1 Preliminary experiments

Preliminary experimentation was performed with a small set of 2 out of 8 instances to avoid overfitting. This selection comprises approximately 25% of the global set and provides enough variability. Notice that 5 rounds are performed for each instance, so each algorithm can reach a maximum of 10 best solutions.

As introduced, the proposed VNS requires an initial solution that has been built using an approach based on greedy construction. Hence, the first experiment is performed to evaluate the impact of using a greedy constructive procedure instead of a random initial solution. Table 3 shows the results obtained when considering one random construction and one greedy construction with the proposed method. In order to verify the relevance of the proposed greedy criterion, an additional constructive method is included in the comparison. Specifically, it is a greedy constructive directly based on the objective function value, i.e., it selects the best node according to the resulting objective function value performing just one iteration. This constructive, named as C_{IDM}, uses the following greedy criterion: $g_{IDM}(u) = AMRIM(S_r \cup u)$. The complexity reported by that method it is $O(|V| \cdot l)$.

The results highlight the relevance of a good constructive method to start from a promising region of the search space. In particular, the $\boldsymbol{C_{RND}}$ procedure is unable to reach any best solution, $\boldsymbol{C_{IDM}}$ obtains one best solution, and $\boldsymbol{C_{GR}}$ achieves 9 of 10 best solutions. Furthermore, the deviations of the procedures $\boldsymbol{C_{RND}}$ and $\boldsymbol{C_{IDM}}$ are drastically large, indicating that they are not even close to the solution found by the $\boldsymbol{C_{GR}}$ procedure. Although the computing time required by $\boldsymbol{C_{GR}}$ is slightly larger than the other ones, the difference is negligible when compared with the increase in quality. It is worth mentioning that the computing time required by $\boldsymbol{C_{GR}}$ of 1.49 s on average is negligible in the context of social network analysis, so we decide to use this constructive procedure in the remaining experiments.

The next experiment is devoted to evaluate the impact of both $\boldsymbol{LS}_{cd}$ and $\boldsymbol{LS}_{sn}$ local search strategies. To this end, Table 4 summarizes the results obtained when applying each local search method to the same initial solution obtained by the constructive procedure. Analyzing the average objective function obtained by both local search procedures compared with the value obtained by the constructive procedure isolatedly (see Table 3), it can be concluded that both

Table 5 Evaluation of the impact of parameter $k_{\max}$ in the BVNS algorithm

$k_{\max}$	Avg.	Dev.	Time	#B
0.1	4268.29	0.04	**19.61**	**8**
0.2	4267.09	0.06	22.79	6
0.3	**4268.66**	**0.03**	31.82	7
0.4	4267.82	0.05	39.36	7
0.5	4266.15	0.08	46.68	6

Best results are highlighted with bold font

local search strategies are able to further improve the initial solution. However, in view of the results, the $\boldsymbol{LS_{cd}}$ strategy outperforms the $\boldsymbol{LS_{sn}}$ strategy. This result can be partially explained, since the $\boldsymbol{LS_{sn}}$ strategy is surrogated by selecting a predefined number δ of nodes in the neighborhood as stated in Lozano-Osorio et al. (2021), thus exploring a narrower portion of the search space.

Furthermore, it is worth mentioning that in 7 out of 10 preliminary instances, better results are obtained using the $\boldsymbol{LS_{cd}}$ strategy. However, in 3 out of 10 preliminary instances, better results are obtained using the $\boldsymbol{LS_{sn}}$ strategy. When designing a local search procedure, the computational effort is one of the key aspects. The results obtained in this experiment, showing that the local search procedure increases the computing time in less than two seconds on average, confirm the appropriate design of the local search method described in Sect. 3.2.

Having defined the constructive procedure and the local search method, it is necessary to fix the parameter of the proposed VNS algorithm. In particular, it requires from a single input parameter $k_{\max}$, the maximum neighborhood to be explored. In order to do so, $k_{\max} = \{0.1 \cdot |S|, 0.2 \cdot |S|, 0.3 \cdot |S|, 0.4 \cdot |S|, 0.5 \cdot |S|\}$ values are tested. Notice that values larger than 0.5 are not included since it would basically consists of reconstructing a solution from scratch, which is not the philosophy of the VNS methodology. These values are included in the final VNS algorithm, considering the $\boldsymbol{C_{GR}}$ as constructive procedure and $\boldsymbol{LS_{cd}}$ as a local search method. Table 5 shows the results obtained in this experiment.

As expected, the computing time increases with the size of $k_{\max}$, which is mainly due to two reasons: i) the maximum neighborhood explored is large, requiring more computing time to explore all the considered neighborhoods and ii) the perturbation size becomes larger with the increase of $k_{\max}$, so the local search method needs to perform a more computationally demanding exploration to reach a local optimum.

Although there are not significant differences among the different values for $k_{\max}$, the results show that the best solutions are found by $k_{\max} = 0.1$, which is the smallest neighborhood considered. It is able to reach the largest number of best solutions while maintaining the second better deviation, really close to the best one (0.04 vs. 0.03), also requiring the smallest computing time. These results are in line with the VNS philosophy Mladenović and Hansen (1997), which indicates that considering small perturbations usually leads to better quality solutions in the VNS metaheuristic. Therefore, $k_{\max} = 0.1$ is selected for the remaining experiments.

4.2 Final experiments

In order to analyze the quality of the proposed algorithm, we perform a competitive testing of the best BVNS according to the preliminary experiments versus the best method found in the state of the art (AdaIMM) according to the recent survey Banerjee et al. (2020) and **IMP** $+\boldsymbol{LS_{sn}}$ Lozano-Osorio et al. (2020, 2024) to be compared with a recent method using machine learning knowledge, considering the complete set of instances. Since the code was made publicly available by the authors,[2] the original results reported by the authors have been successfully replicated in the same computing environment to have a fair comparison. Table 6 shows the results obtained by BVNS, AdaIMM and **IMP** $+\boldsymbol{LS_{sn}}$ over the complete set of instances.

The results depicted in Table 6 show that BVNS is able to reach the best solution in all the instances, while AdaIMM presents a deviation of 24.23%, and, **IMP** $+\boldsymbol{LS_{sn}}$ obtain 21.67%. Analyzing the computational effort of each algorithm, **IMP** $+\boldsymbol{LS_{sn}}$ is the fasted method with 10.23 s, BVNS requires only 27.91 s on average to finish, while AdaIMM needs almost twice the computing time, 53.15 s.

In particular, AdaIMM is showing competitive performance in one instance, Wiki-Vote, which is the smallest one, and also have the largest average out-degree. If we extend this analysis to the remaining instances, we can conclude that the behavior of AdaIMM is better when the average out-degree is large, which are those cases in which it is easier to influence a larger number of users. On the contrary, in those instances with small average out-degree values, BVNS obtains considerably better results, showing AdaIMM a deviation of approximately 40%. Notice that these instances are those in which the process of influencing is more limited, being more challenging for the algorithms.

It is notable that **IMP** $+\boldsymbol{LS_{sn}}$ demonstrates a competitive performance when compared individually to AdaIMM. Specifically, AdaIMM achieves the best solutions in 21 instances, whereas **IMP** $+\boldsymbol{LS_{sn}}$ reaches the best solutions in 19 instances. It is important to highlight that AdaIMM requires approximately five times more computational time.

[2] https://github.com/lichao-sun/Multi-Round-Influence-Maximization

Table 6 Results obtained by BVNS, AdaIMM and **IMP +LS_{sn}** algorithms

		BVNS				AdaIMM				IMP +LS_{sn}			
SN	R	Avg.	Dev.	Time	#B	Avg.	Dev.	Time	#B	Avg.	Dev.	Time	#B
CA AstroPh	1	9647.01	0.00	34.35	1	5784.98	40.03	100.14	0	9625.07	0.23	8.17	0
	2	9667.75	0.00	52.20	1	6483.23	32.94	111.85	0	9629.74	0.39	16.42	0
	3	9692.10	0.00	66.38	1	8053.73	16.90	152.20	0	9631.27	0.63	24.63	0
	4	9713.66	0.00	119.49	1	8069.84	16.92	153.87	0	9647.68	0.68	32.92	0
	5	9720.14	0.00	161.17	1	8551.72	12.02	155.74	0	9653.96	0.68	40.69	0
CA CondMat	1	5111.51	0.00	8.45	1	3219.52	37.01	35.08	0	5074.97	0.71	2.49	0
	2	5126.04	0.00	11.20	1	3912.49	23.67	55.76	0	5082.57	0.85	5.08	0
	3	5165.89	0.00	18.58	1	4145.90	19.74	59.08	0	5108.87	1.10	7.76	0
	4	5181.80	0.00	31.42	1	4303.16	16.96	60.37	0	5118.37	1.22	10.50	0
	5	5215.02	0.00	64.88	1	4667.92	10.49	65.19	0	5140.85	1.42	13.16	0
Cit HepPh	1	2392.71	0.00	4.97	1	1753.39	26.72	20.15	0	1135.18	52.56	0.66	0
	2	3081.24	0.00	6.55	1	2464.14	20.03	28.68	0	1662.73	46.04	1.69	0
	3	3450.72	0.00	9.79	1	2747.52	20.38	25.66	0	2049.26	40.61	3.25	0
	4	3860.18	0.00	15.61	1	2955.17	23.44	27.83	0	2297.85	40.47	4.77	0
	5	4097.83	0.00	24.60	1	3358.45	18.04	34.37	0	2559.08	37.55	6.51	0
Email Enron	1	9591.72	0.00	17.23	1	5107.23	46.75	82.69	0	9569.04	0.24	12.50	0
	2	9621.66	0.00	28.22	1	6050.43	37.12	95.61	0	9594.68	0.28	23.85	0
	3	9657.68	0.00	43.96	1	6499.42	32.70	103.29	0	9596.08	0.64	33.45	0
	4	9670.71	0.00	67.86	1	7274.42	24.78	117.35	0	9617.37	0.55	41.58	0
	5	9697.83	0.00	126.99	1	7632.77	21.29	119.02	0	9618.30	0.82	49.12	0
Flixster	1	5731.22	0.00	8.62	1	5461.67	4.70	52.16	0	5021.51	12.38	2.19	0
	2	7940.12	0.00	13.43	1	6157.46	22.45	71.98	0	6070.13	23.55	5.93	0
	3	8996.23	0.00	23.21	1	7103.62	21.04	73.39	0	7054.71	21.58	9.58	0
	4	9812.33	0.00	40.05	1	8018.47	18.28	88.11	0	7610.67	22.44	13.58	0
	5	10570.99	0.00	67.28	1	8481.25	19.77	95.04	0	8023.87	24.10	18.12	0
NetHEPT	1	90.23	0.00	0.14	1	58.99	34.62	0.35	0	38.06	57.82	0.05	0
	2	153.38	0.00	0.29	1	107.82	29.70	0.66	0	69.86	54.45	0.12	0
	3	203.81	0.00	0.59	1	153.12	24.87	0.78	0	97.85	51.99	0.17	0
	4	252.35	0.00	1.13	1	191.02	24.30	1.16	0	121.63	51.80	0.24	0
	5	291.05	0.00	1.95	1	221.91	23.76	1.20	0	146.41	49.70	0.30	0
Wiki-Vote	1	1725.87	0.00	2.21	1	1703.09	1.32	23.29	0	1692.43	1.94	1.14	0
	2	1765.88	0.00	3.38	1	1724.59	2.34	23.27	0	1718.30	2.69	2.22	0
	3	1794.89	0.00	6.12	1	1737.57	3.19	26.31	0	1751.70	2.41	3.55	0
	4	1818.00	0.00	10.49	1	1753.37	3.56	30.82	0	1787.83	1.66	4.76	0
	5	1834.54	0.00	16.05	1	1766.00	3.74	30.29	0	1798.37	1.97	6.04	0
p2p Gnutella31	1	98.63	0.00	0.23	1	50.49	48.81	0.36	0	54.36	44.88	0.15	0
	2	171.32	0.00	0.54	1	93.33	45.52	0.57	0	85.76	49.94	0.30	0
	3	235.64	0.00	1.13	1	125.50	46.74	0.66	0	108.82	53.82	0.44	0
	4	296.92	0.00	2.13	1	157.55	46.94	0.74	0	134.63	54.66	0.58	0
	5	356.54	0.00	3.68	1	194.67	45.40	1.03	0	158.74	55.48	0.72	0
Summary		4837.58	0.00	27.91	40	3707.42	24.23	53.15	0	4373.96	21.67	10.23	0

Best results are highlighted with bold font

Finally, both methods exhibit a deviation of 9.81% for **IMP +LS_{sn}** and 9.80% for AdaIMM.

Another important conclusion that can be derived from the results is that the computing time required to finish increases with the number of rounds. This behavior is partially explained, since the number of nodes to be selected also increases with the number of rounds. However, the proposed method shows good scalability with a linear increase in the computing time with respect to the number of rounds.

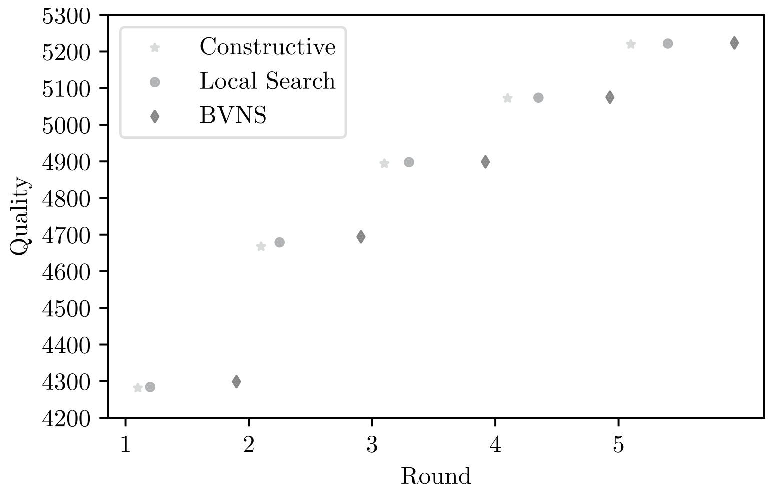


Fig. 6 Comparison between the Avg. for all instances in each component of the proposal algorithm

In order to confirm the statistically significance of these results, the pairwise Wilcoxon Signed Rank Test has been conducted, obtaining a p-value < 0.001. Therefore, it can be ensured that there are significant differences between the results obtained by both algorithms, emerging BVNS as a competitive algorithm for solving the AMRIM.

To conclude, it is interesting to analyze the contribution of each phase of the final BVNS algorithm, in order to detect which are the key parts of the algorithm or, even more, if there is any phase which is not contributing to the final results. Figure 6 shows the contribution desegregated per each phase of the proposal (greedy method, local search and BVNS) to evaluate the effect of each phase in the final results. The graph is divided by rounds (x-axis), indicating the quality (y-axis) obtained by each phase in each round.

The division between the different phases illustrates a direct relationship between the duration of each phase and their respective complexity. Notably, the constructive method has a constant computational time in each round due to its greedy nature, while the local search shows an increase in computational time with the growing number of rounds, since more possible moves can be performed. Furthermore, it is observed that the BVNS stands out as the most computationally demanding method in terms of time. This can be partially explained since, inside BVNS, several executions of the local search are performed, increasing its complexity.

If we now analyze the quality of the solutions obtained in each phase, it is seen how BVNS is able to reach the best results. Notice that the local search emerges as a good alternative to generate high-quality solutions when the time available is restricted. Finally, it is worth mentioning that finding an improvement becomes a more challenging task with the increase of the number of rounds.

The results obtained show that BVNS is a competitive method for AMRIM compared to the best method found in the literature, obtaining solutions better than those for AdaIMM in an efficient way.

5 Conclusions

In this paper, a Basic Variable Neighborhood Search algorithm for solving the AMRIM is presented. A comparison of three constructive methods is carried out, with the constructive method using a greedy criterion performing best. Additionally, a local search based on communities has been proposed, obtaining better results than a recent work on a surrogated local search. The proposed BVNS is able to outperform the best method found in the literature, named AdaIMM, obtaining new best-known solutions in every considered instance by requiring half of the computing time (27.91 s vs. 53.15 s). In order to compare with recent methods, the method $\mathbf{IMP} + \boldsymbol{LS}_{sn}$ has been included in the analysis, no obtaining any best-known solution with a deviation (21.67% where AdaIMM has 24.23%) and requiring less than 5 times the computational time of AdaIMM.

The main drawback in these methods is that they need to perform a complete evaluation, due to the demand that requires Monte Carlo simulation. However, an efficient implementation leads us to provide high-quality solutions in reasonable computing times, even for the largest instances derived from real-world SNs commonly considered in SNI problems. This fact makes the proposed AMRIM algorithm highly scalable.

The primary challenges in this family of problems are related to scalability, particularly in large networks where the computational effort can become prohibitive, as well as the accuracy of the IDM due to its probabilistic nature. Our proposal obtains high quality solutions in short computing time, using efficient metaheuristic algorithms that are scarce in the area. Nevertheless, it can be further improved by new methods able to either provide more quality or reduce the complexity. These limitations promote the ongoing research for new realistic approaches in this field.

As a future work, it was found interesting to adapt the 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. New parallelism techniques thanks to GPU can provide less computational time, and other community detection problems could obtain other interesting nodes selection.

Acknowledgements The authors acknowledge support from the Spanish Ministry of "Ciencia, Innovación y Universidades" (MCIN/AEI/10.13039/501100011033/FEDER, UE) under grant ref. PID2021-126605NB-I00 and PID2021-125709OA-C22, the "Comunidad de Madrid", "Fondos Estructurales" of the European Union with grant reference S2018/TCS-4566, and Universidad Rey Juan Carlos with "Proyectos Impulso", project name "Análisis de Redes Sociales Avanzado".

Author contributions I.L.O. developed the code for the algorithm and wrote an initial draftJ.S.O. and A.D found fundings, designed the algorithmic proposal and review the initial draft of the manuscript.

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflicts of interest The authors declare no conflict of interest.

References

Aghaee Z, Ghasemi MM, Beni HA, Bouyer A, Fatemi A (2021) A survey on meta-heuristic algorithms for the influence maximization problem in the social networks. Computing 103(11):2437–2477. https://doi.org/10.1007/s00607-021-00945-7

Aghaee Z, Kianian S (2020) Influence maximization algorithm based on reducing search space in the social networks. SN Appl Sci 2(12):2067. https://doi.org/10.1007/s42452-020-03812-w

Arepalli P, Narayana V, Venkatesh R, Kumar N (2019) Certified node frequency in social network using parallel diffusion methods. Ingénierie des Systèmes d'Inf 24(1):113–117. https://doi.org/10.18280/isi.240117

Banerjee S, Jenamani M, Pratihar DK (2020) A survey on influence maximization in a social network. Knowl Inf Syst 62(9):3417–3455. https://doi.org/10.1007/s10115-020-01461-4

Barbieri N, Bonchi F, Manco G (2012) Topic-aware social influence propagation models. In: 2012 IEEE 12th International Conference on Data Mining, pp. 81–90. 10.1109/ICDM.2012.122

Berger J (2014) Word of mouth and interpersonal communication: a review and directions for future research. J Consum Psychol 24(4):586–607. https://doi.org/10.1016/j.jcps.2014.05.002

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Bouyer A, Ahmadi Beni H, Arasteh B, Aghaee Z, Ghanbarzadeh R (2023) Fip: A fast overlapping community-based influence maximization algorithm using probability coefficient of global diffusion in social networks. Expert Syst Appl 213:118869. https://doi.org/10.1016/j.eswa.2022.118869

Bucur D, Iacca G (2016) Influence maximization in social networks with genetic algorithms. In: Applications of evolutionary computation: 19th european conference, evoapplications 2016, Porto, Portugal, March 30–April 1, 2016, Proceedings, Part I 19. Springer, pp 379–392. https://doi.org/10.1007/978-3-319-31204-0_25

Chen W, Peng B, Schoenebeck G, Tao B (2022) Adaptive greedy versus non-adaptive greedy for influence maximization. J Artif Intell Res 74:303–351. https://doi.org/10.1613/jair.1.12997

Chen W, Wang C, Wang Y (2010) Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '10, pp. 1029–1038. Association for Computing Machinery, New York, NY, USA. 10.1145/1835804.1835934

Chen N (2008) On the approximability of influence in social networks. In: Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '08, pp. 1029–1037. Society for Industrial and Applied Mathematics, USA. 10.5555/1347082.1347195

Chunaev P (2020) Community detection in node-attributed social networks: A survey. Comput Sci Rev 37:100286. https://doi.org/10.1016/j.cosrev.2020.100286

Duarte A, Pantrigo JJ, Pardo EG, Mladenović N (2014) Multi-objective variable neighborhood search: an application to combinatorial optimization problems. J Global Optim 63(3):515–536. https://doi.org/10.1007/s10898-014-0213-z

D'angelo A, Agarwal A, Jin KX, Juan YF, Klots L, Moskalyuk O, Wong Y (2009) Targeting advertisements in a social network. Google Patents. US Patent App. 12/195,321

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826

Golovin D, Krause A (2010) Adaptive submodularity: theory and applications in active learning and stochastic optimization. Journal of Artificial Intelligence Research 42:427. https://doi.org/10.48550/arXiv.1003.3967

Gong M, Song C, Duan C, Ma L, Shen B (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, Lu W, Lakshmanan LVS (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, ???. 10.1145/1963192.1963217

Granovetter M (1978) Threshold models of collective behavior. Am J Sociol 83(6):1420–1443

Hansen P, Mladenović N, Pérez JAM (2009) Variable neighbourhood search: methods and applications. Ann Oper Res 175(1):367–407. https://doi.org/10.1007/s10479-009-0657-6

Hansen P, Mladenović N, Brimberg J, Pérez JAM (2010) Variable Neighborhood Search, pp. 61–86. Springer, Boston, MA. 10.1007/978-1-4419-1665-5_3

Herrán A, Colmenar JM, Duarte A (2019) A variable neighborhood search approach for the vertex bisection problem. Inf Sci 476:1–18. https://doi.org/10.1016/j.ins.2018.09.063

Hinz O, Skiera B, Barrot C, Becker JU (2011) Seeding strategies for viral marketing: an empirical comparison. J Mark 75(6):55–71. https://doi.org/10.1509/jm.10.0088

Hosni AIE, Li K (2020) Minimizing the influence of rumors during breaking news events in online social networks. Knowl Based Syst 193:105452. https://doi.org/10.1016/j.knosys.2019.105452

Huang K, Tang J, Han K, Xiao X, Chen W, Sun A, Tang X, Lim A (2020) Efficient approximation algorithms for adaptive influence maximization. VLDB J 29(6):1385–1406. https://doi.org/10.1007/s00778-020-00615-8

Jaouadi M, Ben Romdhane L (2024) A survey on influence maximization models. Expert Syst Appl 248:123429. https://doi.org/10.1016/j.eswa.2024.123429

Kempe D, Kleinberg J, Tardos E (2015) Maximizing the spread of influence through a social network. Theory Comput 11(1):105–147. https://doi.org/10.4086/toc.2015.v011a004

Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146. 10.1145/956750.956769

Khalil E, Dilkina B, Song L (2013) Cuttingedge: Influence minimization in networks. In: Proceedings of Workshop on Frontiers of Network Analysis: Methods, Models, and Applications at NIPS, pp. 1–13. Citeseer

King SF, Burgess TF (2008) Understanding success and failure in customer relationship management. Ind Mark Manage 37(4):421–431. https://doi.org/10.1016/j.indmarman.2007.02.005

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

Lawyer G (2015) Understanding the influence of all nodes in a network. Sci Rep 5(1):8665. https://doi.org/10.1038/srep08665

Leskovec J, Krause A, Guestrin C, Faloutsos C, VanBriesen J, Glance N (2007) Cost-effective outbreak detection in networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 420–429. 10.1145/1281192.1281239

Li Y, Gao H, Gao Y, Guo J, Wu W (2023) A survey on influence maximization: from an ml-based combinatorial optimization. ACM Trans Knowl Discov Data 17(9):1–50. https://doi.org/10.1145/3604559

Liu X, Li M, Li S, Peng S, Liao X, Lu X (2014) Imgpu: Gpu-accelerated influence maximization in large-scale social networks. IEEE Trans Parallel Distrib Syst 25(1):136–145. https://doi.org/10.1109/TPDS.2013.41

Liu Y, Wang X, Kurths J (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

Liu X, Li S, Liao X, Wang L, Wu Q (2012) In-time estimation for influence maximization in large-scale social networks. In: Proceedings of the Fifth Workshop on Social Network Systems. SNS '12. Association for Computing Machinery, New York, NY, USA. 10.1145/2181176.2181179

Lozano-Osorio I, Martínez-Gavara A, Martí R, Duarte A (2022) Max-min dispersion with capacity and cost for a practical location problem. Expert Syst Appl 200:116899. https://doi.org/10.1016/j.eswa.2022.116899

Lozano-Osorio I, Sanchez-Oro J, Rodriguez-Garcia MÁ, Duarte A (2020) Optimizing computer networks communication with the band collocation problem: a variable neighborhood search approach. Electronics 9(11):1860. https://doi.org/10.3390/electronics9111860

Lozano-Osorio I, Sánchez-Oro J, Duarte A (2023) An efficient and effective grasp algorithm for the budget influence maximization problem. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-023-04680-z

Lozano-Osorio I, Sánchez-Oro J, Duarte A, Cordón Ó (2021) A quick GRASP-based method for influence maximization in social networks. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-021-03510-4

Lozano-Osorio I, Sánchez-Oro J, Duarte A, Sörensen K (2024) What characteristics define a good solution in social influence minimization problems? In: Metaheuristics international conference. Springer, pp 328–333. https://doi.org/10.1007/978-3-031-62922-8_23

Luo C, Cui K, Zheng X, Zeng D (2014) Time critical disinformation influence minimization in online social networks. 2014 IEEE Joint Intelligence and Security Informatics Conference, 68–74 10.1109/JISIC.2014.20

Malliaros FD, Rossi M-EG, Vazirgiannis M (2016) Locating influential nodes in complex networks. Sci Rep 6(1):19307

Martín R, Cavero S, Lozano Osorio I (2022) rmartinsanta/mork: v0.13. Zenodo . 10.5281/ZENODO.6671107

Mladenović N, Hansen P (1997) Variable neighborhood search. Comput Op Res 24(11):1097–1100. https://doi.org/10.1016/S0305-0548(97)00031-2

Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113

Nguyen Hung T, Thai My T, Dinh Thang N (2016) Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In: Proceedings of the 2016 International Conference on Management of Data. SIGMOD '16, pp. 695–710. Association for Computing Machinery, New York, NY, USA. 10.1145/2882903.2915207

Pérez-Peló S, Sánchez-Oro J, Gonzalez-Pardo A, Duarte A (2021) A fast variable neighborhood search approach for multi-objective community detection. Appl Soft Comput 112:107838. https://doi.org/10.1016/j.asoc.2021.107838

Pérez-Peló S, Sánchez-Oro J, Martín-Santamaría R, Duarte A (2019) On the analysis of the influence of the evaluation metric in community detection over social networks. Electronics 8(1):23. https://doi.org/10.3390/electronics8010023

Ravelo SV, Meneses CN (2021) Generalizations, formulations and subgradient based heuristic with dynamic programming procedure for target set selection problems. Comput Op Res 135:105441. https://doi.org/10.1016/j.cor.2021.105441

Richardson M, Domingos P (2002) Mining knowledge-sharing sites for viral marketing. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 61–70. 10.1145/775047.775057

Shakarian P, Bhatnagar A, Aleali A, Shaabani E, Guo R (2015) The Independent Cascade and Linear Threshold Models, pp. 35–48. Springer, Cham. 10.1007/978-3-319-23105-1_4

Stanley, W., Katherine, F.: Social Network Analysis. Cambridge University Press, ??? (1994). 10.1017/cbo9780511815478

Sun L, Huang W, Yu PS, Chen W (2018) Multi-Round Influence Maximization (Extended Version). arXiv. 1048550/ARXIV.1802.04189

Sun L, Huang W, Yu PS, Chen W (2018) Multi-round influence maximization. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2249–2258. 10.1145/3219819.3220101

Sánchez-Oro J, Pantrigo JJ, Duarte A (2014) Combining intensification and diversification strategies in VNS. an application to the vertex separation problem. Comput Op Res 52:209–219. https://doi.org/10.1016/j.cor.2013.11.008

Tang Y, Shi Y, Xiao X (2015) Influence maximization in near-linear time: A martingale approach. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1539–1554

Vaswani S, Lakshmanan LVS (2016) Adaptive Influence Maximization in Social Networks: Why Commit when You can Adapt? arXiv.1048550/ARXIV.1604.08171

Wang C, Chen W, Wang Y (2012) Scalable influence maximization for independent cascade model in large-scale social networks. Data Min Knowl Disc 25(3):545–576. https://doi.org/10.1007/s10618-012-0262-1

Wang S, Jin Y, Cai M (2023) Enhancing the robustness of networks against multiple damage models using a multifactorial evolutionary algorithm. IEEE Trans Syst, Man, and Cybern Syst 53(7):4176–4188. https://doi.org/10.1109/tsmc.2023.3241621

Wang S, Liu W (2023) Enhancing the robustness of influential seeds towards structural failures on competitive networks via a memetic algorithm. Knowl Based Syst 275:110677. https://doi.org/10.1016/j.knosys.2023.110677

Wang Y, Wang H, Li J, Gao H (2016) Efficient influence maximization in weighted independent cascade model. In: Navathe, S.B., Wu, W., Shekhar, S., Du, X., Wang, S.X., Xiong, H. (eds.) Database Systems for Advanced Applications, pp. 49–64. Springer, Cham. 10.1007/978-3-319-32049-6_4

Wu X, Fu L, Zhang Z, Long H, Meng J, Wang X, Chen G (2020) Evolving influence maximization in evolving networks. ACM Trans Internet Technol 20(4):1–31. https://doi.org/10.1145/3409370

Yuste J, Pardo EG, Duarte A (2023) Variable neighborhood descent for software quality optimization. In: Di Gaspero, L., Festa, P., Nakib, A., Pavone, M. (eds.) Metaheuristics, pp. 531–536. Springer, Cham. 10.1007/978-3-031-26504-4_44