A Multiobjective Evolutionary Approach for Influence Maximization in Multilayer Networks

Qipeng Lu
Nanjing University of Finance and Economics
Nanjing, 210023, Jiangsu, China
qipenglu@qq.com

Zhan Bu*
Nanjing University of Finance and Economics
Nanjing, 210023, Jiangsu, China
buzhan@nuaa.edu.cn

Yuyao Wang
Nanjing University of Science and Technology
Nanjing, 210094, Jiangsu, China
wangyuyao@njust.edu.cn

* corresponding author.

ABSTRACT

Influence Maximization (IM) is one key algorithmic problems in information diffusion research; it aims to select a set of users from a social network and, by following a specific model, maximize the number of users influenced (the influence spread). Yet despite its immense potential, relatively little research is dedicated to IM for multilayer networks. Conversely, most existing IM studies that rely on a greedy algorithm strategy only obtain a single solution that provides limited insights on the target networks' core organization. With that in mind, we focus on studying the Influence Maximization Problem (IMP) in multilayer networks. Specifically, we define novel concepts, such as the pairwise reciprocal length and pairwise influence, with respect to the information-diffusion process in multilayer networks. Then we formulate the IM in multilayer networks as a multiobjective optimization problem and employ the classic Nondominated Sorting Genetic Algorithm II (NSGA-II) to find a set of Pareto-optimal solutions that provide a wide range of options for decision makers. To maintain population diversity and accelerate the algorithm's convergence, we combine a heuristic population initialization strategy and an efficient two-point crossover operation. Extensive experiments show that our approach has competitive performance when compared to off-the-shelf IM algorithms with regard to influence spread and running time.

CCS Concepts

• **Networks→Network types→Overlay and other logical network structures→Online social networks.**

Keywords

Influence maximization; multilayer networks; multiobjective optimization; NSGA-II; pareto optimal.

1. INTRODUCTION

Online social networks' rapid development has made it easy for people to share information and communicate. In this context, many large-scale online social network platforms (OSNPs) with more than billions of users have emerged, including Twitter, Facebook and Pinterest. Usually, these OSNPs not only serve as communication tools for users, but also as potential marketing mediums for companies and advertisers. Furthermore, compared with traditional marketing tools, marketing based on OSNPs is characterized by high profits and low investment. However, because each company's advertising budget is limited, one feasible solution is to find a small set of advertising agents (for example, online users) who can trigger a massive influence on OSNPs. Formally, we formulate this as the Influence Maximization Problem (IMP) in the field of complex network analysis-that is, mining a certain number of seed nodes that can maximize the influence spread under certain influence diffusion models.

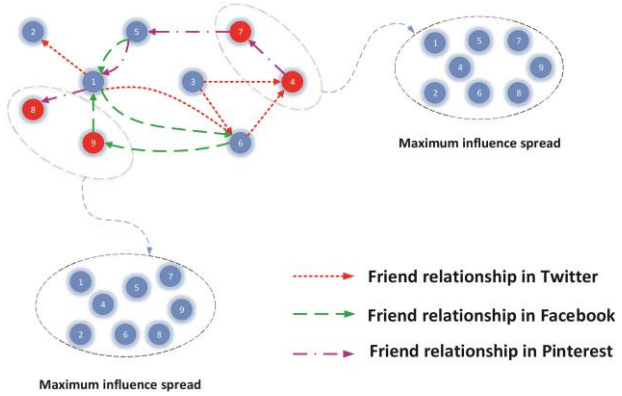


Figure 1. A hypothetical example of real-world social networks, where each node denotes a social actor and the directed links of red, green and purple represent their friendships on Twitter, Facebook and Pinterest, respectively.

Existing information diffusion research has given considerable attention to the IM problem. However, typically social actors are not confined to one OSNP—they participate in different OSNPs. One such scenario involves multiple online users (social actors) maintaining several social accounts simultaneously, allowing them to propagate information across different OSNPs. Figure 1 shows a hypothetical example, where each node denotes a social actor, and the directed links of red, green and purple represent their friendships on Twitter, Facebook, and Pinterest, respectively. For example, Actor 1 receives a piece of information from her Twitter friend, she can share this information with friends on Twitter, Facebook, and Pinterest. In this example, we show this by using three accounts and then transferring information on three OSNPs. Once the actor's information is exposed to her friends, it further spreads on the three OSNPs. If we only focus on one OSNP, it can lead to an inaccurate estimate of how the

ICCAI '20, April 23–26, 2020, Tianjin, China.

ACM ISBN 978-1-4503-7708-9/20/04…$15.00
DOI: https://doi.org/10.1145/3404555.3404568

information spreads, as this approach fails to identify the most influential users. So instead, we study this as an IM problem in multilayer networks, where users' influence is evaluated based on their participation in all the OSNPs. Considering this from another perspective, if we select two users as seed nodes and propagate information through them, multiple approaches could yield a maximum influence spread. For example, when Actors 4 and 7 are selected to propagate information, this can lead to a maximum influence spread, comprised of eight users receiving the information. However, selecting Actors 8 and 9 as the seed nodes could yield the same results. It is interesting that most existing IM studies rely on a Greedy algorithm strategy and only obtain a single solution; this motivates us to explore multiple solutions that provide a wide range of options for decision makers.

With this in mind, here we focus on studying the IM problem in multilayer networks. First, we build a multilayer network based on the interlayer relationships of different networks. Then, we define some novel concepts, such as the pairwise reciprocal length and pairwise influence, which visually reveal the information diffusion process in multilayer networks. Finally, we formulate IM in multilayer networks as a multiobjective optimization problem [2]. Specifically, we construct a multiobjective optimization model, which considers the centrality and information propagation capability of the candidate seed nodes. To solve the target problem, we use a genetic algorithm to explore the broad search space of all possible sets of seed nodes. The multiobjective genetic algorithm's purpose is to find a set of optimal solutions in a single run and provide a range of choices for decision makers. Thus, we propose a novel IM algorithm based on Nondominated Sorting Genetic Algorithm (NSGA-II) [8]. To maintain population diversity, and accelerate the algorithm's convergence, we combine a heuristic population initialization strategy and an efficient two-point crossover operation.

The rest of this paper is organized as follows: Section 2 briefly describes related work on IM in social networks. Section 3 defines and considers the IMP in multilayer networks. Section 4 details our approach. In Section 5, we thoroughly evaluate our approach and its performance using many experiments. Finally, we conclude in Section 6.

2. RELATED WORK

The study of IM in social networks has become a hot research topic with the advance of the internet and smart devices. Researchers are trying to figure out how to maximize the spread of influence in networks by mining a certain number of seed nodes. In the literature, many attentions have been directed to this problem, which follow different strategies, as elaborated below.

The problem of IM was first proposed and studied by Domingos and Richardson et al [9]. Next, Kempe et al., formulated the IM problem as a discrete optimization problem and proved that IM is a nondeterministic polynomial time-hard optimization problem [13]. They also proposed three widely used information diffusion models-Linear Threshold (LT), Independent Cascade (IC), and Weight Cascade (WC)-and proposed the Hill-Climbing Greedy algorithm with a (1-1/e)- approximation of optimal influence spread [17]. This greedy method requires a sufficient number of Monte Carlo simulations to be run on information diffusion models to obtain accurate estimations of average influence spread. Thus, as the scale of the network continues to expand, the application of this greedy algorithm is limited, and it is difficult to meet large-scale social networks. Determining how to reduce the algorithm's computational time is a key challenge.

To improve the greedy algorithm's efficiency, some new greedy algorithms have been proposed. Leskovec et al. [15] proposed a lazy greedy algorithm called Cost-Effective Lazy Forward selection (CELF), which reduces the number of simulations by mining the submodular of the influence function. The method is nearly 700 times faster than the hill-climbing greedy algorithm. In addition, Goyal et al. [10] proposed CELF++ to further optimize CELF; experimental results show that CELF++ improves performance by nearly 35~55 percent compared to CELF. Although these improved greedy algorithms improve the runtime, they have poor scalability for large-scale networks.

To address the scalability issue, Chen et al. proposed several heuristic methods, including Degree Discount [5] and PMIA [4], to approximate the influence propagation using each node's local arborescence structures. Jung et al. [12] proposed the IRIE algorithm, which integrates the advantages of influence ranking (IR) and influence estimation (IE) methods for IM. Although these heuristic algorithms are quite efficient, their accuracy can be much lower than a greedy algorithm. As we mentioned previously, because the IM is NP-hard, these methods based on greedy and heuristic algorithms cannot efficiently find promising solutions; instead, they find the optimal set only under certain conditions and some level of approximation.

In recent years, some approaches attempted to tackle the IM problem by means of computational intelligence, exploiting methods such as Simulated Annealing [11] and Evolutionary Algorithms [3]. Evolutionary algorithms were found capable of effectively exploring the vast search space of all possible subsets of nodes. In particular, the genetic algorithms do not require any assumptions about the graph underlying the network, and more feasible solutions are available in these algorithms than current heuristics. These approaches show some promising results on relatively large datasets obtained from real-world networks.

3. NOTATION DEFINITION AND PROBLEM FORMULATION

We can represent a multilayer network as a two-tuple, i.e., **M**=<**V**,**W**>, where $\mathbf{V} = \{i: i \in \{1, \cdots, n\}\}$ is the set of n entities. $\mathbf{W} = \{W^{\alpha}: \alpha \in \{1, \cdots, L\}\}$ is the family of L weighted adjacency matrices, where each element, i.e., $W^{\alpha} = (w_{i,j}^{\alpha}) \in \mathbb{R}^{n \times n}$, represents the directed and weighted network without self-loops on the α th layer of **M**. In this paper, $\forall \alpha \in \{1,2, \cdots, L\}$ and $\forall i, j \in \mathbf{V}$, we assume that $w_{i,j}^{\alpha} \in [0,1]$, representing the probability that entity *i* transmits the information upon contacting with entity *j* on the α th layer. In this way, the set of out-neighbors associated with entity *i* on the αth layer is defined as $\widehat{N}_i^{\alpha} = \{j^{\alpha}: w_{i,j}^{\alpha} > 0\}$. In contrast with the classic mathematical framework defined by Domenico et al., the interlayer spreading processes in **M** are created only by the same entity, who has participated in two different layers. In other words, a particular piece of information can spread from one layer to another, through the same entity. To emphasize entity *i*'s position on the α layer, we sometimes represent it as i^{α}. Moreover, $\forall \alpha, \beta \in \{1,2, \cdots, L\}$, $\alpha \neq \beta$ and $\forall i \in \mathbf{V}$, the inter-layer spreading probability from i^{α} to i^{β} is fixed to be 1. We also comprehend that each entity in **V** has the ability of context-awareness on multiple layers of **M**. In a realistic setting, a particular entity *i* can be viewed as a real person; note that *i* can have multiple social network accounts, such as Facebook and Twitter. If *i* receives a piece of information from

his Facebook friend *j*, *i* can send the same information to his other friend *k* on Twitter.

Understanding how information spreads in multilayer networks is an important problem, having implications for both predicting the size of epidemics, as well as for planning effective interventions. One of the important ideas with regard to the spreading processes in multilayer networks is that information can also spread from one layer to another. The set of spreading paths from entities *i* to *j* over the multilayer network **M** (denoted by $SetPath_{i\to j}$) is defined as

$$\begin{aligned} SetPath_{i\to j} = \{\left(v_0^{\beta_0}, v_1^{\alpha_1,\beta_1}, \cdots, v_\chi^{\alpha_\chi}\right) \mid_{i=v_0 \wedge j=v_\chi}: \\ \forall\tau \in \{0,\cdots,\chi\}, \forall_\tau \in \mathbf{V}, \\ \forall\tau,\tau' \in \{0,\cdots,\chi\},\ v_\tau \neq v_{\tau'}, \\ \forall\tau \in \{1,\cdots,\chi\},\ \beta_{\tau-1} = \alpha_\tau \wedge w^{\alpha_\tau}_{v_{\tau-1},\ v_\tau} > 0\} \end{aligned} \quad (1)$$

where $w^{\alpha_\tau}_{v_{\tau-1},v_\tau}$ indicates the probability that entity $v_{\tau-1}$ transmits the information upon contacting his/her out-neighbor entity v_τ on the α_τth layer. Here, the spreading path (if any) from entities *i* to *j* must be acyclic, and each entity $v_\tau(\tau \in \{1,\cdots,\chi\})$ on such a path can be chosen as a random out-neighbor of entity $v_{\tau-1}$ on any arbitrary layer. Thus, this path's probability is given by

$$pro_{i\to j}\left(v_0^{\beta_0}, v_1^{\alpha_1,\beta_1}, \cdots, v_\chi^{\alpha_\chi}\right) = \prod_{\tau=1}^{\chi} w^{\alpha_\tau}_{v_{\tau-1},v_\tau} \quad (2)$$

where χ denotes the path's length. Then, the pairwise reciprocal length from *i* to *j* is defined as the reciprocal of the minimal length of all spreading paths from *i* to *j*:

$$rel_{i\to j} = \max_{\left(v_0^{\beta_0}, v_1^{\alpha_1,\beta_1,\cdots,v_\chi^{\alpha_\chi}}\right)\in SetPath_{i\to j}} \frac{1}{\chi} \quad (3)$$

and the pairwise influence of *i* on *j* is defined as the maximal probability of all spreading paths from *i* to *j*:

$$inf_{i\to j} = \max_{\left(v_0^{\beta_0}, v_1^{\alpha_1,\beta_1,\cdots,v_\chi^{\alpha_\chi}}\right)\in SetPath_{i\to j}} pro_{i\to j}(v_0^{\beta_0}, v_1^{\alpha_1,\beta_1}, \cdots, v_\chi^{\alpha_\chi}) \quad (4)$$

Broadly speaking, the higher the values of $rel_{i\to j}$ and $inf_{i\to j}$, the shorter the distance and the greater the influence, from entities *i* to *j* over the multilayer network **M**, respectively. Note that if there is no spreading path from *i* to *j* (i.e., $SetPath_{i\to j} = \emptyset$), we have $rel_{i\to j} = inf_{i\to j} = 0$.

For a given multilayer network **M**=<**V**, **W**>, let $\mathcal{A}^{proj} = (a_{i,j}^{proj}) \in \mathbb{R}^{n\times n}$ denote the project network's adjacency matrix, such that $a_{i,j}^{proj} = 1$ if $\exists\alpha \in \{1,2,\cdots,L\}$, $w_{i,j}^{\alpha} > 0$ and $a_{i,j}^{proj} = 0$ otherwise. Then, the projection network's corresponding weighted adjacency matrix is $\mathcal{W}^{proj} = (w_{i,\ j}^{proj}) \in \mathbb{R}^{n\times n}$, where $w_{i,j}^{proj} = \max_{\alpha\in\{1,2,\cdots,L\}} w_{i,j}^{\alpha}$. Along these lines, we calculate the reciprocal length matrix $\mathcal{M}^R = (rel_{i\to j}) \in \mathbb{R}^{n\times n}$ and the influence matrix $\mathcal{M}^I = (inf_{i\to j}) \in \mathbb{R}^{n\times n}$ by employing Dijkstra-like algorithms on $\mathcal{A}^{proj}$ and $\mathcal{W}^{proj}$, respectively. Note that by using an entity-based structure, each algorithm's time complexity achieves $O(n^2 \log n)$; because of space constraints, we omit the detailed algorithms here.

In this work, we denote the social influence graph associated with **M** as $\mathcal{P} = (p_{i,\ j}) \in \mathbb{R}^{n\times n}$, where each element $p_{i,j}$ indicates the degree of entity *i*'s influence on entity *j*. Specifically, we define $p_{i,\ j} = w_{i,j}^{proj}/\varpi_j$, where $\varpi_j = \sum_{i=1}^{n} w_{i,j}^{proj}$ is the in-strength of entity *j* in the projection network. Then, we have $\sum_{i=1}^{n} p_{i,j} = 1$ for any entity *j* in **V**. During the influence propagation process, each entity is either active or inactive. Let $\boldsymbol{s^t} = (s_1^t, \cdots, s_n^t) \in \Omega$ denote the binary state vector at time *t*, where $s_i^t = 1$ if entity *i* is active at time *t*, and $s_i^t = 0$ otherwise; $\Omega = \{0,1\}^{1\times n}$ is the solution space of $\boldsymbol{s^t}$. Based on the Linear Threshold (LT) model, active entities will remain active forever and an inactive entity *j* will become active at time *t*+1 if

$$\sum_{i=1}^{n} s_i^t p_{i,j} > \theta \quad (5)$$

where $0 < \theta < 1$ is the randomly selected activation threshold. Because the set of entities is finite, the state transition process will stop in finite iterations *T*, such that $\boldsymbol{s^T} = \boldsymbol{s^{T+1}}$. Simply put, let $\mathbf{x} = (x_1, x_2, \cdots, x_n) = \boldsymbol{s^0}$ denote the initial state vector; we sometimes write **x** as $(x_i, \mathbf{x}_{-i})$ to emphasize entity *i*'s initial state, where $\mathbf{x}_{-i} = (x_1, \cdots, x_{i-1}, x_{i+1}, x_n)$ indicates the initial states of other entities except *i*. The expected number of active entities when propagation stops, also called the influence spread of **x**, is denoted by $\sigma(\mathbf{x})$. Along these lines, the IMP is to find an initial state vector **x** that can achieve maximum $\sigma(\mathbf{x})$.

Another classical propagation model is the Independent Cascade (IC) model. In the model, an entity's state in a social network is either active or inactive, and active entities influence their inactive neighbors. An entity's state can switch from being inactive to active, but not vice versa. The model has a parameter called propagation probability *p*, which models individuals' tendency to be affected by its neighbors. IC's diffusion mechanism can be described as follows. The diffusion process begins with an initial state vector of active entities **x** at round $T = 0$. Let $\mathbf{x} = (x_1, x_2, \cdots, x_n) = \boldsymbol{s^0}$. At each round *T*, an active entity x_i from the last round $\boldsymbol{s^{T-1}}$ will be given a single chance to influence each of its inactive neighbors x_j, with a propagation probability *p*. If x_j is influenced, it is activated and added to set $\boldsymbol{s^T}$. The process terminates when $\boldsymbol{s^T}$ is empty. The set of entities influenced by the initial state vector **x** is the union of $\boldsymbol{s^T}$ generated at each round. Under the IC model, we also denote the number of entities influenced by the initial state vector **x** as $\sigma(\mathbf{x})$.

Because it usually costs to inject or promote information to members of **x**, it is important to maximize the initial state vector $\mathbf{x}'$ potential utility (in this case, its influence-spreading ability) into consideration:

$$f^C(\mathbf{x}) = \sum_{j:x_j=0} \max_{i:x_i=1} rel_{i\to j} \quad (6)$$

$$f^A(\mathbf{x}) = \sum_{j:x_j=0} \max_{i:x_i=1} inf_{i\to j} \quad (7)$$

where $f^C(\mathbf{x})$ and $f^A(\mathbf{x})$ denote the centrality and accessibility associated with the initial state vector **x**. It is easy to check that both $f^C(\cdot): \Omega \to \mathbb{R}_0^+$ and $f^A(\cdot): \Omega \to \mathbb{R}_0^+$ are monotone submodular functions. Thus, we can employ the greedy optimization (that is, Algorithm 1) to find the (1-1/e)-approximation for the problem of $\max_{\mathbf{x}\in\Omega} f^C(\mathbf{x})$ or $\max_{\mathbf{x}\in\Omega} f^A(\mathbf{x})$ subject to $\sum_{i=1}^{n} x_i = K$ (see Theorem of Nemhauser, Wolsey, Fisher [16]). Instead of finding the optimal **x** for each single-objective optimization problem, the IMP in this paper is formulated as a multiobjective optimization problem (MOOP) [18]:

$$\begin{aligned} &\max_{\mathbf{x}\in\Omega} \left(f^C(\mathbf{x}), f^A(\mathbf{x})\right), \\ &s.t.,\ g(\mathbf{x}) = \sum_{i=1}^{n} x_i = K \end{aligned} \quad (8)$$

where $g(\cdot)$: $\Omega \to \mathbb{R}_0^+$ is the constraint function. In other words, we focus our attention on finding a set of K seed entities that achieve maximum$(f^C(\mathbf{x}), f^A(\mathbf{x}))$. However, because the two objectives in Eq.(8) could conflict with each other, it is difficult (or impossible) to find a single solution that optimizes both objectives simultaneously. An alternative strategy is to balance the two objectives so that each one has a relatively satisfied value.

Next, we introduce a novel genetic algorithm (GA) based on the classic NSGA-II [8] to find the set of Pareto-optimal solutions with regard to the MOOP [19] in Eq.(8).

Algorithm 1: Greedy optimization

Require: $\mathcal{M}^R, \mathcal{M}^I, K, f(\cdot) \in \{f^C(\cdot),\ f^A(\cdot)\}$

Ensure: The initial state vector **x;**

1. $\mathbf{x} \leftarrow 0^n$;
2. **for** k=1: K **do**
3. select $i = \arg\max_{j:x_j=0}(f(x_j,\ \mathbf{x}_{-j})_{x_j=1} - f(\mathbf{x}))$;
4. $x_i \leftarrow 1$;
5. **end for**
6. **return x**;

4. METHODOLOGY

4.1 Initializing Population

During the process of population initialization, instead of choosing seed entities from **V**, we employ a heuristic approach to obtain a candidate seed set. The main assumption is that an entity with a higher centrality (i.e., $R_i = \sum_{j\neq i} rel_{i\to j}$) and greater influence (i.e., $I_i = \sum_{j\neq i} inf_{i\to j}$) could be more convenient for propagating information in the multilayer network. Based on this assumption, we combine the fast nondominated sorting with the crowding distance assignment to sort entities in **V**; and then select the top 10 percent of entities in the sorted sequence to obtain the candidate seed set **CS**. Next, we randomly select K entities from **CS** to generate an initial state vector x; by repeating this process N times, we create an initial population of size N. Algorithm 2 provides the full pseudocode.

Algorithm 2: Population initialization

Require: $\mathcal{M}^R, \mathcal{M}^I, K, N$

Ensure: The candidate seed set and the initial population;

1. **for** $i = 1{:}n$ **do**
2. $R_i \leftarrow \sum_{j\neq i} rel_{i\to j}$;
3. $I_i \leftarrow \sum_{j\neq i} inf_{i\to j}$;
4. **end for**
5. fast non-dominated sorting (**V**);
6. crowing distance assignment (**V**);
7. sort entities in **V** according to the dominated level (upward) and the crowing distance (downward);
8. **CS**← the top 10% entities in the sorted **V**;
9. **for** $o = 1{:}N$ **do**
10. $\mathbf{x}^{(o)}$ ← randomly select K entities from **CS** and mark them as active;
11. $\boldsymbol{P^0}.add(\mathbf{x}^{(o)})$;
12. **end for**
13. **return CS** and $\boldsymbol{P^0}$;

4.2 Genetic Operators

The purposes of genetic operators are to maintain the solution populations diversity, and speed up two objectives' convergence to the Pareto optimality. To achieve these two goals, the proposed IMA-NSGA-II combines and uses two kinds of genetic operators, namely crossover and mutation.

Crossover is the process of exchanging portions of two chromosomes to produce new chromosomes, and it is one of the key factors in natural biological evolution. We cannot apply traditional crossover operators directly, such as one-point, partial-mapped, cycle and uniform crossovers, to solve the MOOP in Eq.(8), because this does not fully satisfy the constraint $g(\mathbf{x}) = K$. To address this issue, we propose a modified two-point crossover operator (see Figure 2). Specifically, given two solutions $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, we randomly select two entities, i.e., i and j, such that $x_i^{(1)} = 1, x_j^{(1)} = 0, x_i^{(2)} = 0$ and $x_j^{(2)} = 1$. Then, by switching the alleles on $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, we obtain two new solutions, $\mathbf{x}^{(1*)}$ and $\mathbf{x}^{(2*)}$, such that $\forall v \notin \{i,\ j\}, x_v^{(1*)} = x_v^{(1)}, x_v^{(2*)} = x_v^{(2)}, x_j^{(1*)} = x_i^{(2*)} = 1$ and $x_i^{(1*)} = x_j^{(2*)} = 0$. Obviously, this two-point crossover operator strictly guarantees that the constraint equation in Eq.(8) is always true, while simultaneously maintaining the solution population's diversity.

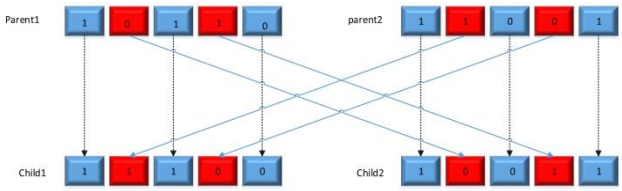


Figure 2. An example of the two-point crossover operation in IMA-NSGA-II.

To further expedit the convergence of the two objectives to Pareto optimality, we apply the stochastic hill-climbing strategy to IMA-NSGA-II. The proposed mutation operator contains two main substeps: first, we remove one active entity if decreasing one objective makes it the smallest; second, we add another inactive entity from the candidate seed set **CS** under the condition that increasing the opposite objective makes it the largest. To be specific, for a given solution $\mathbf{x}^{(3)}$, we randomly choose a function $\{f^C(\cdot),\ f^A(\cdot)\}$, and thus the opposite function is denoted by $f(\cdot)$. Then, an active entity i will become inactive if

$$i = \arg\min_{j:x_j=1}\left(f(\mathbf{x}^{(3)}) - f\left(x_j^{(3)},\ \mathbf{x}_{-j}^{(3)}\right)_{x_j^{(3)}=0}\right) \quad (9)$$

and at the same time, an inactive entity j will become active if

$$j = \arg\max_{i:x_i=0\wedge i\in CS}\left(f'\left(x_i^{(3)},\ \mathbf{x}_{-i}^{3}\right)_{x_i^{(3)}=1} - f'(\mathbf{x}^{(3)})\right) \quad (10)$$

thus, we obtain a new solution $\mathbf{x}^{(3*)}$, such that $\forall v \notin \{i,\ j\},\ x_v^{(3*)} = x_v^{(3)},\ x_i^{(3*)} = 0$ and $x_j^{(3*)} = 1$. It is also easy to check that after the aforementioned mutation operator, the new solution $\mathbf{x}^{(3*)}$ is not dominated by the original solution $\mathbf{x}^{(3)}$, such that $(f^C(\mathbf{x}^{(3)}),\ f^A(\mathbf{x}^{(3)})) \prec (f^C(\mathbf{x}^{(3*)}),\ f^A(\mathbf{x}^{(3*)}))$. This property can accelerate the convergence of the two objectives to Pareto optimality.

4.3 Implementing IMA-NSGA-II

In the past few decades, researchers have tried to solve MOOPs using evolutionary algorithms (EAs). NSGA-II, proposed by Deb et al., stands as one of the most representative EAs, which can converge closer to the true Pareto-optimal front and maintains a nice population diversity. In addition to NSGA-II, there are other noteable EAs, such as NSGA-III [7], generalized differential evolution (GDE3) [14], and multiobjective evolutionary algorithm based on decomposition (MOEA/D). We refer the interested reader to [3] for a state-of-the-art introduction on EAs.

Here, the proposed IM algorithm is realized under the framework of NSGA-II, where each solution (also called a chromosome in EAs) is viewed as an initial-state vector. Similar to NSGA-II, it employs a fast non-dominated sorting method to evaluate all chromosomes in a population with a satisfactory speed, which generates nondominated sets with different levels. Then it harnesses crowding-distance assignment to maintain a relatively dispersed distribution of chromosomes. For details on fast nondominated sorting and crowding-distance assignment, please refer to Deb et al.'s work [8]. Algorithm 3 describes the details of IMA-NSGA-II. In line 1, we perform Algorithm 2 to generate an initial population P^0 with N chromosomes. From lines 2 to 3, the fast nondominated sorting and crowding-distance assignment are used in combination to calculate every chromosome's crowding distance at each level of the nondominated sets. Then, each iteration of IMA-NSGA-II (between line 4 and 23) contains three steps. First, our solution performs a series of genetic operators (see Section 4.2) to generate the tth offspring population (see lines 5-11). Second, it performs fast nondominated sorting on the union set of P^t and Q^t, to generate nondominated sets with different levels (lines 12-13). Third, it sorts all chromosomes in $P^t \cup Q^t$ based on the nondominant level and crowding distance, and we obtain the t+1th population by choosing the top N chromosomes in the sorted sequence (lines 14-22). IMA-NSGA-II repeats this procedure $\mathbb{T}$ times, and thus the nondominant chromosomes in $P^{\mathbb{T}}$ will be viewed as the multiresolution solutions of MOOP in Eq.(8).

Table 1. Two datasets' basic statistics

Datasets	TK&TP	TO&TD
Nodes	2683	1501
Edges	7916	4573
Avg. Degree	5.901	6.093
Avg. Weighted Degree	0.761	0.782
Avg. Path Length	8.089	10.719
Avg. Clustering Coefficient	0.424	0.431

Algorithm 3: The general framework of IMA-NSGA-II

Require: $\mathcal{M}^R, \mathcal{M}^I, K, N, f(\cdot) \in \{f^C(\cdot),\ f^A(\cdot)\}, \mathbb{T}$;

Ensure: A set of non-dominated solutions in **PF**;

1. $[\boldsymbol{CS},\ \boldsymbol{P^0}] \leftarrow$ population initialization using Algorithm 2;
2. $\boldsymbol{F} \leftarrow$ fast non-dominated sorting ($\boldsymbol{P^0}$)
3. $\forall \boldsymbol{F_I} \in \boldsymbol{F}$, crowing distance assignment ($\boldsymbol{F_I}$);
4. **for** $t = 1:\mathbb{T}$ **do**
5. $\boldsymbol{Q^t} \leftarrow \emptyset$;
6. **while** $|\boldsymbol{Q^t}| \leq |\boldsymbol{P^t}|$ **do**
7. $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\} \leftarrow$binary tournament selection ($\boldsymbol{F}$);
8. $\{\mathbf{x}^{(1*)},\ \mathbf{x}^{(2*)}\} \leftarrow$ crossover operator ($\mathbf{x}^{(1)},\ \mathbf{x}^{(2)}$);
9. $\mathbf{x}^{(3*)} \leftarrow$ mutation operator ($\mathbf{x}_1,\ \boldsymbol{CS}$);
10. $\boldsymbol{Q^t} \leftarrow \boldsymbol{Q^t} \cup \{\mathbf{x}^{(1*)},\ \mathbf{x}^{(2*)},\ \mathbf{x}^{(3*)}\}$;
11. **end while**
12. $\boldsymbol{R^t} \leftarrow \boldsymbol{P^t} \cup \boldsymbol{Q^t}$;
13. $\boldsymbol{F'} \leftarrow$ fast non-dominated sorting ($\boldsymbol{R^t}$);
14. $\boldsymbol{P^{t+1}} \leftarrow \emptyset$ and $\boldsymbol{I} \leftarrow 1$;
15. **while** $\left|\boldsymbol{P^{t+1}}\right| + |\boldsymbol{F'_I}| \leq |\boldsymbol{P^t}|$ **do**
16. crowding distance assignment ($\boldsymbol{F'_I}$);
17. $\boldsymbol{P^{t+1}} \leftarrow \boldsymbol{P^{t+1}} \cup \boldsymbol{F'_I}$;
18. $\boldsymbol{I} \leftarrow \boldsymbol{I} + \mathbf{1}$;
19. **end while**
20. crowding distance assignment ($\boldsymbol{F'_I}$);
21. Sort $\boldsymbol{F'_I}$ in descending order of the crowing distance;
22. $\boldsymbol{P^{t+1}} \leftarrow \boldsymbol{P^{t+1}} \cup \boldsymbol{F'_I}[\mathbf{1}: (|\boldsymbol{P^t}| - |\boldsymbol{P^{t+1}}|)]$;
23. **end for**
24. **return** $\boldsymbol{P}^{\mathbb{T}}$;

5. EXPERIMENTS

We evaluate IMA-NSGA-II on two real-world datasets, which are multilayer cooperative networks extracted from the well-known DBLP service platform. We conduct a series of experiments to compare its performance with state-of-the-art baseline methods.

5.1 Experimental Setup

To obtain real-world multilayer networks, we crawled five years published articles (from 2013 to 2018) in the following IEEE journals: IEEE Transactions on Knowledge and Data Engineering (TKDE), IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), IEEE/ACM Transactions on Networking (TON) and IEEE Transactions on Dependable and Secure Computing(TDSC) from DBLP, respectively. Using the five-year publication data of each journal, we construct a cooperative network, where each node (entity) represents an author. If two authors *i* and *j* have co-authored at least one article, in which *i*'s ranking is ahead of *j*'s, there is a directed link from *i* to *j*. Moreover, the weight on the directed link (*i,j*) is measured by $CoArticle(i,j)/Article(i)$, where $CoArticle(i,j)$ indicates the number of articles co-authored by *i* and *j*, given that *i*'s ranking is ahead of *j*'s, and *Article(i)* represents the total number of articles authored by *i*. Along these lines, we obtain four cooperative networks, denoted as W^{TKDE}, W^{TPAMI}, W^{TON}, and W^{TDSC}, respectively. For multilayer networks, we build through multiple cooperative networks. Specifically, we only create the interlayer connections of a multilayer network dataset *TK&TP* using the same entities, that participated in W^{TKDE} and W^{TPAMI}. Likewise, we obtain another dataset (*TO&TD*). Here, we focus on analyzing and studying IM in multilayer networks. For convenience of calculation, we map the multilayer network to the projection network and the weight on the directed link from *i* to *j* is represented by the maximum $w^{\alpha}_{i,j}$. Moreover, to more accurately reveal the process of information dissemination, we extract the maximal connected subgraphs from each dataset and study their IMP. Table 1 offers basic statistics about the two datasets.

In this paper, we compare the IMA-NSGA-II with several classic algorithms, including the Greedy, High Degree [17], Degree Discount [5], PageRank [1] and LDAG [6]. The following is a list of algorithms we evaluate in our experiments.

- **LDAG**: This algorithm is designed for the LT model. We use the influence parameter $\theta = \frac{1}{320}$ to control the local DAG's size constructed for each node.

- **Degree Discount**: This is a basic degree discount heuristic algorithm applicable to all cascade models, with a propagation probability of p=0.1. Although this heuristic is designed specifically for the independent cascade model, we use it as a general heuristic in the class of degree heuristics. It performs much better than the pure-degree heuristics.
- **High Degree**: As a kind of comparison algorithm, High Degree is a simple heuristic algorithm for selecting seed sets. The main idea is that the higher the degree, the more influential the node. It selects top-k nodes with the highest out-degree as a seed set.
- **Greedy**: This is a greedy algorithm that uses lazy-forward optimization [15]. To obtain an accurate estimate of influence spread, we run 10,000 simulations.
- **PageRank**: This popular algorithm rank webpages. The weight $w_{j,i}$ on the edge from entity i to entity j indicates the transition probability. Intuitively, $w_{j,i}$ indicates the influence strength of entity j to entity i, and we use it in the opposite direction as a "vote" of entity i to entity j, as explained by the PageRank algorithm. K nodes with the highest PageRanks will be selected as seed sets. We set the restart probability for PageRank as 0.15, and the stop criterion as 0.0001 in L_1 norm.

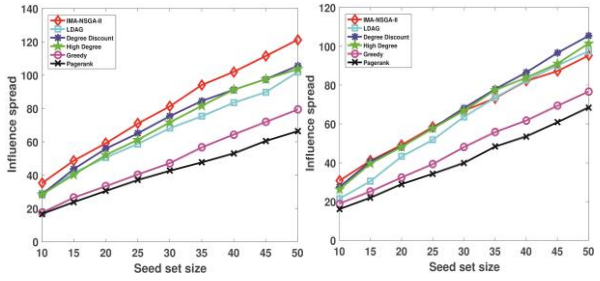
To further evaluate our proposed algorithm's robustness and compare the algorithms' accuracy, we use the IC and LT methods to compute the influence spread of each algorithm's final solution. Following previous work, for the IMA-NSGA-II algorithm we proposed, some parameters are set to N=100, $\mathbb{T} = 500$, and K ranges from 10 to 50. To obtain the influence spread of all the algorithms, for each seed set we run the simulation of the IC or LT model on the networks 10,000 times. Moreover, since our two networks are sparsely directed, the propagation probability is set to 0.1 in the IC model.

5.2 Experimental Results

5.2.1 Influence Spread for Real-World Datasets

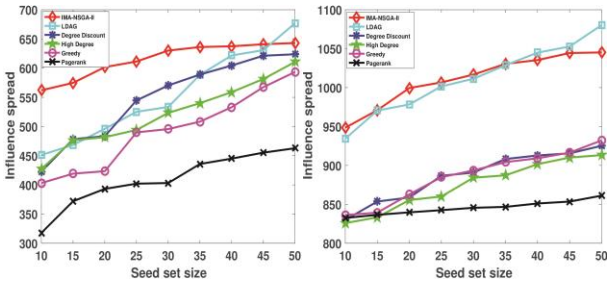
Because our algorithm is an evolutionary algorithm, we can get a set of solutions for different seed set sizes. To facilitate a comparison for different seed set sizes, we choose a solution with a large influence spread. This demonstrates the advantages of our proposed algorithm, which yields many solutions in one run, and provides more strategies for decision makers.

As we know, in the IC model, the propagation probability of entities plays an important role in influence spread. Because our datasets are sparse and directed, we set the propagation probability of entities to 0.1. The seed size K ranges from 10 to 50. For ease of reading, all the percentage differences reported below on influence spreads are for the case of K=50. Figure 3 reports the influence spread of various algorithms on the two datasets under the IC model. In Figure 3(a), we intuitively see that the algorithm proposed by us always has an absolute advantage over other algorithms in terms of the influence spread. Compared with other comparison algorithms, IMA-NSGA-II algorithm is 18.97, 14.92, 17.07, 52.83, and 82.72 percent better than LDAG, Degree Discount, High Degree, Greedy, and PageRank respectively. Additionally, as the number of seed entities increase, Degree Discount and High Degree have similar performances but are significantly worse than the IMA-NSGA-II algorithm's. Besides, in Figure 3, there is an obvious phenomenon where the LDAG algorithm is always inferior to the Degree Discount and High Degree algorithms, because the LDAG algorithm is designed specifically for the LT model, and is unsuitable for the IC model. In Figure 3(b), we observe that when K is less than 30, the IMA-NSGA-II algorithm outperforms all other comparison algorithms. But, as the number of seeds increases, the growth rate of the influence spread changes slowly for the IMA-NSGA-II algorithm, and it gradually loses its competitive edge. PageRank (as the baseline) performs quite badly; it is 28.08, 29.76, 35, 32.49, and 10.64 percent lower than IMA-NSGA-II, LDAG, Degree Discount, High Degree, and Greedy algorithms in its influence spread achieved on the *TO&TD* dataset, respectively.



(a) TK&TP network (b) TO&TD network

Figure 3. Various algorithms' influence spread under the independent cascade model with propagation probability p=0.1.



(a)TK&TP network (b) TO&TD network

Figure 4. Various algorithms' influence spread under the linear threshold model.

To further verify our algorithm's effectiveness and robustness, we conducted a large number of experiments based on the LT model. Figure 4 reports the influence spread of various algorithms on the two datasets under the LT model. In Figure 4(a), for IMA-NSGA-II algorithms, there is an obvious phenomenon that the increase rate of influence spread changes slowly after K reaches 20. Moreover, the IMA-NSGA-II algorithm's performance on influence spread is better than LDAG between K=10 and 35. However, with K increasing, IMA-NSGA-II performs 3.25 percent worse than LDAG. When K is greater than 35, the seed set selected by LDAG has the maximum influence spread among all the algorithms on the *TK&TP* dataset. Additionally, the Degree Discount algorithm always performs better than High Degree and PageRank, and it is fairly close to the Greedy algorithm on the *TK&TP* dataset. We also observe that when the seed set's size is greater than 15, PageRank's pergormance is the worst-it is 17.61, 20.29, 6.92, 5.7, and 7.61 less than the influence spread of the IMA-NSGA-II, LDAG, Degree Discount, High Degree, and Greedy algorithms, respectively. In Figure 4(b), the IMA-NSGA-II algorithm has a competitive advantage for all algorithms, and its performance is worse than the LDAG algorithm when the K is greater than 45.

5.2.2 *Running Time for Real-World Datasets*

To demonstrate the difference in computational effectiveness between our proposed algorithm and others, we set up experiments to compare running times. Figure 5 reports various algorithms' running time for selecting 50 seeds on the two real-world datasets. From Figure 5, we see that the Greedy algorithm is the most computationally expensive; it takes thousands of seconds on both the *TK&TP* and *TO&TD* datasets. As expected, when the dataset size increases, Greedy's running time also increases. Moreover, our IMA-NSGA-II algorithm selects 50 seed nodes on both the *TK&TP* dataset and *TO&TD* dataset under a stable computational time. Specifically, the algorithm takes a few seconds to select 50 seed nodes and is more than two to three orders of magnitude faster than Greedy. We also observe that, for the two real-world datasets, the IMA-NSGA-II algorithm is more computationally expensive than the state-of-the-art LDAG. This is because IMA-NSGA-II spans a broad search of all possible subsets of nodes and finds a set of optimal solutions in one run, which is quite time consuming. By comparison, other algorithms (PageRank, Degree Discount, and High Degree) have low running times that outperform IMA-NSGA-II. In terms of running time, High Degree has the best performance, but it cannot find a seed set with good quality. Greedy selects reliable solutions, but the algorithm offers poor scalability as the network becomes larger and more complex. Hence, we conclude that because IMA-NSGA-II balances good efficiency with reliable solutions, it is the best choice to seek more feasible solutions in one run.

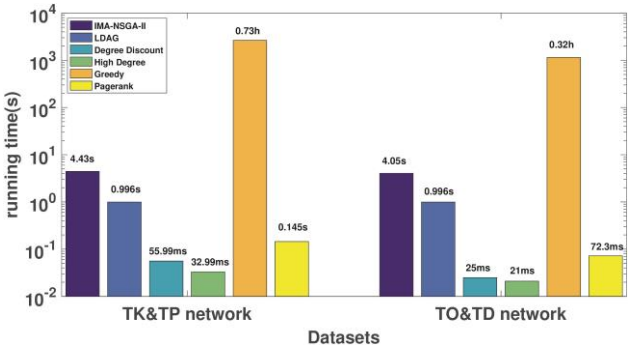


Figure 5. Running time of different algorithms on the two datasets.

6. CONCLUSION

In this paper, we introduced a novel multi-objective evolutionary approach IMA-NSGA-II for IM in multilayer networks. We performed extensive experiments on two real-world datasets. The heuristic algorithms are in general more efficient time-wise(depending on the dataset and size of the seed nodes). In short, our experiments showed that on each dataset the IMA-NSGA-II algorithm's performance is at least comparable to that of the best-tested algorithm. However, because LDAG and heuristic algorithms rely on specific network topological features, we observe that their performance significantly varies from one model to another. On the other hand, because the IMA-NSGA-II does not require any assumptions about the graph under the network, its results are more stable in the two datasets. Overall, our experiments revealed that EAs are viable tools to solve the IMP, especially when seeking more than one feasible solution.

7. ACKNOWLEDGMENTS

This research was partially supported by National Natural Science Foundation of China under Grant 71871109, Grant 71801123, and Grant 91646204, and Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant KYCX18-1440.

8. REFERENCES

[1] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.

[2] Zhan Bu, Huijia Li, Chengcui Zhang, and et al. 2019. Graph K-means based on leader identification, dynamic game and opinion dynamics. *IEEE Transactions on Knowledge and Data Engineering* 1, 1 (2019), 1–14.

[3] Doina Bucur, Giovanni Iacca, Andrea Marcelli, Giovanni Squillero, and Alberto Tonda. 2018. Improving Multi-objective Evolutionary Influence Maximization in Social Networks. In *International Conference on the Applications of Evolutionary Computation*. Springer, 117–124.

[4] Wei Chen, Chi Wang, and Yajun Wang. 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*. ACM, 1029–1038.

[5] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 199–208.

[6] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *2010 IEEE International Conference on Data Mining*. IEEE, 88–97.

[7] Kalyanmoy Deb and Himanshu Jain. 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems with Box Constraints. *IEEE Transactions on Evolutionary Computation* 18, 4 (2014), 577–601.

[8] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[9] Pedro Domingos and Matt Richardson. 2001. Mining the Network Value of Customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 57–66.

[10] Amit Goyal, Wei Lu, and Laks VS Lakshmanan. 2011. CELF++: optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th International Conference Companion on World Wide Web*. ACM, 47–48.

[11] Qingye Jiang, Guojie Song, Cong Gao, Wang Yu,Wenjun Si, and Kunqing Xie. 2011. Simulated Annealing Based Influence Maximization in Social Networks. In *AAAI Conference on Artificial Intelligence.* ACM

[12] Kyomin Jung, Wooram Heo, and Wei Chen. 2012. IRIE: Scalable and robust influence maximization in social networks. In *2012 IEEE 12th International Conference on Data Mining.* IEEE, 918–923.

[13] David Kempe, Jon Kleinberg, and Éva Tardos. 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. ACM, 137–146.

[14] Saku Kukkonen and Jouni Lampinen. 2005. GDE3: The third evolution step of generalized differential evolution. In *2005 IEEE Congress on Evolutionary Computation*, Vol. 1. IEEE, 443–450.

[15] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos F-aloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 420-429.

[16] G. L. Nemhauser, L.A. Wolsey, and M.L. Fisher. 1978. An analysis of approximations for maximizing submodular set function. *Mathematical Programming,* 14,1 (1978), 265-294.

[17] E. Tardos, D. Kempe, and J. Kleinberg. 2003. Maximizing the spread of influence in a social network. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 137–146.

[18] A. Hamdy and A. A. Mohamed. 2019. Greedy binary particle swarm optimization for multi-objective constrained next release problem. *International Journal of Machine Learning and Computing.* 9.5(2019), 63-69.

[19] Liansong Xu and Dazhi Pan. 2018. Multi-objective optimization based on chaotic particle swarm optimization. *International Journal of Machine Learning and Computing.* 8.3(2018), 128-139.