Applied Soft Computing 136 (2023) 110062

Contents lists available at ScienceDirect

Applied Soft Computing

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

ABEM: An adaptive agent-based evolutionary approach for influence maximization in dynamic social networks

Weihua Li [a,*], Yuxuan Hu [b], Chenting Jiang [b], Shiqing Wu [b,c], Quan Bai [b], Edmund Lai [a]

[a] School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand
[b] School of Information and Communication Technology, University of Tasmania, Hobart, Tasmania, 7005, Australia
[c] School of Computer Science, University of Technology Sydney, Sydney, New South Wales, 2007, Australia

ARTICLE INFO

Article history:
Received 6 October 2021
Received in revised form 14 January 2023
Accepted 18 January 2023
Available online 2 February 2023

Keywords:
Influence maximization
Evolutionary computing
Genetic algorithm
Agent-based modeling

ABSTRACT

Influence maximization is recognized as a crucial optimization problem, which aims to identify a limited set of influencers to maximize the coverage of influence dissemination in social networks. However, real-world social networks are usually dynamic and large-scale, which leads to difficulty in capturing real-time user and diffusion features to effectively and accurately select the key influencers. In this paper, we propose an adaptive agent-based evolutionary approach to address this challenging issue with agent-based modeling and genetic algorithm. This novel approach identifies the users' influence capability in a distributed manner and optimizes the influencer set selection in a dynamic environment. An adaptive solution optimizer is proposed as one of the key components, driving the evolutionary process and adapting the candidate solutions dynamically. The proposed approach is also applicable to large-scale networks due to its distributed framework. Evaluation of our approach is performed by using both synthetic networks and real-world datasets. Experimental results demonstrate that the proposed approach outperforms state-of-the-art seeding algorithms in terms of maximizing influence.

1. Introduction

In recent years, influence diffusion modeling and maximization in complex networks such as online social networks have attracted a great deal of attention from both researchers and practitioners. It has applications in many areas, including decision-making, marketing, and social influence analysis [1–6]. On the other hand, influence maximization is also considered as a controversial technology, which has been applied to spreading rumors or fake news [7,8], manipulate public opinion [9,10] and even suppress minorities. Motivated by this background, many research works have been dedicated to modeling the user characteristics and simulating the influence process to maximize positive social influence and minimize the negative impact [11,12].

Influence maximization is crucial for optimizing the information diffusion to achieve the maximum influence in a social network [13], for instance, extending the targeted market and winning political campaigns [14]. It is usually achieved by identifying a small number of users who are capable of spreading their influence quickly and widely, thus maximizing the impact across entire the social network [1,3]. However, influence maximization has been proven as an NP-hard and represented as a combinatorial optimization problem [1]. Moreover, the process of influencer mining, known as seed selection, is very challenging in dynamic and large-scale networks to obtain a set of influencers, known as a seed set. For example, in different structures of social networks, such as a random network [15] or a scale-free network [15,16], the influence growth will be based on various network models, such as a configuration model [17] and a Barabási–Albert (BA) preferential attachment model [16], to increase the network scale dynamically. This is because users join and quit, the relationships form and vanish, and the strength of these relationships varies over time, leading to the topology of the network evolving continuously in real-world social networks [18,19].

Based on the traditional influence diffusion models, such as the Independent Cascade (IC) model and the Linear Threshold (LT) model [1], most existing influence maximization algorithms cannot achieve high-efficient and continuous optimization of seed selection within dynamic and large-scale social networks. For example, several classical greedy-based algorithms, such as the greedy algorithm [1], the CELF algorithm [20], the CELF++ algorithm [21], demonstrate a low time efficiency, especially for large-scale networks [22,23], and are inapplicable for dynamic social networks.

* Corresponding author.
E-mail addresses: weihua.li@aut.ac.nz (W. Li), yuxuan.hu@utas.edu.au (Y. Hu), chenting.jiang@utas.edu.au (C. Jiang), shiqing.wu@uts.edu.au (S. Wu), quan.bai@utas.edu.au (Q. Bai), edmund.lai@aut.ac.nz (E. Lai).

https://doi.org/10.1016/j.asoc.2023.110062

Meanwhile, some heuristic algorithms, such as Random heuristic [1], Degree heuristic [24], Pagerank heuristic [25,26], discrete particle swarm optimization (DPSO) [27,28], ant colony optimization (ACO) [29], and differential evolution (DE) [30], cannot guarantee a fast and exact seed choosing process simultaneously and to be suitable in both dynamic and large-scale social networks [22,23]. Furthermore, most community-based algorithms, such as the Community-Based Influence Maximization (CIM) algorithm [31] and the Detecting Influential Nodes (DIN) algorithm [32], are impotent to handle temporal information of community's features or present a high time complexity in large-scale social networks [33]. Therefore, Agent-Based modeling (ABM) and Genetic Algorithm (GA) are leveraged to deal with the influence maximization problem since ABM is acknowledged as one of the appropriate tools for simulating individual behaviors and Evolutionary Computation (EC) techniques have been widely adopted to address the optimization problems [34].

ABM, also called individual-based modeling, has shown its superior in modeling complex systems [35–37]. The ABM is a specific individual-based computational model, where individuals are modeled the interactive autonomous agents [38]. Different from traditional centralized models, ABM is acknowledged as an appropriate approach to explore the macro world by defining the micro level of a social system [35,36]. ABM has been applied to modeling and simulating social influence diffusion, where the diffusion process is presented as an evolutionary process driven by individuals' behaviors [38–40].

GA, as one of the typical evolutionary computation approaches, has been utilized in place of traditional heuristic methods due to three major benefits [22,23,34,41–43]. First, GA provides flexible search and optimization strategies for adapting to a series of complex network conditions [34]. Second, evolutionary metaheuristics in GA facilitates an efficient searching process in a well-defined problem space, which incorporates a massive number of encoded candidate solutions, referred to as seed sets in the influence maximization problem [34]. Third, continuously evolving multiple seed sets in GA not only guarantees the diversity of algorithm solutions but also naturally optimizes the quality of algorithm solutions [23,41]. However, the existing GAs for influence maximization does not have the competency to capture the dynamic topological information.

To improve and extend GA's ability for handling dynamic large-scale social networks, we propose a novel model, called the Adaptive Agent-Based Evolutionary Model (ABEM), by integrating GA and ABM to tackle the influence maximization problem. In the proposed model, we leverage the ABM to model each individual as a proactive agent which can explore its influence capacity in real-time. According to the automatic influence evaluation and nomination mechanism of each agent, the ABM provides a preliminary optimization process by generating an influencer pool. The setting of the influencer pool is beneficial for narrowing down the search space of GA and collecting up-to-date information from dynamic networks. Furthermore, we build an adaptive solution optimizer to transfer the dynamic information of the influence pool from ABM to GA. The optimizer is capable of retrieving the updated influencer candidates from ABM and re-calibrating the potential candidates with the evolution of GA. Finally, we develop a variant of GA to initiate influencer mining based on potential candidate solutions. The outcomes exhibit a sequence of continuously real-time influence maximization solutions in dynamic and large-scale social networks.

Accordingly, the advantages of our proposed model can be demonstrated in three aspects. **First**, ABEM presents a strong optimization ability since a two-level optimal process initiated by using ABM and GA can guarantee the quality of the seed set to maximize influence. **Second**, ABEM significantly improves the time efficiency of mining influencers by distributing the computational cost of the automatic influence evaluation and nomination to each agent. It also narrows down the search space of GA and constantly re-calibrates potential solutions based on the changing environment. **Third**, ABEM can simultaneously reflect the properties of large scale and dynamics in real social networks. This is mainly due to the nature of ABM and the adaptive optimizer component in a distributed framework. Finally, we evaluate the performance of ABEM by conducting four experiments, including convergence analysis, continuous influence maximization in dynamic networks, classical influence maximization comparison and parameter analysis. The experimental results demonstrate that ABEM not only outperforms the state-of-the-art algorithms in the performance of influence maximization but also can be applied to a dynamic and large-scale environment.

To summarize, our main contributions of this research work are listed as follows.

- We first utilized a distributed approach by improving GA to address the influence maximization problem in online social networks.
- We proposed a novel framework by combining GA and ABM to optimize the seed selection from two levels, i.e. the individual level and the global level.
- We developed a novel approach, which can handle large-scale social networks by distributing the major computational cost to the individuals but retaining the optimization process in a central component.
- We developed a novel mechanism for mining influencers with adaptation capabilities, which can handle the fast-changing environment of online social networks.

The rest of this paper is organized as follows. In Section 2, we review the related work. The formal definitions and problem formulation are given in Section 3. In Section 4, we demonstrate the overall process and provide a detailed description of the proposed ABEM approach. Experiments and the analysis of experimental results are presented in Section 5. The paper is concluded in Section 6.

2. Related work

2.1. Classical influence maximization

Influence Maximization Problem (IMP) was initially formulated as a discrete combinatorial optimization problem by Kempe et al. [1,44–46]. They proposed a basic greedy algorithm that offered an approximate guarantee for optimizing the seed set selection based on the IC and LT models [1]. However, the greedy algorithm inefficiently handles large-scale and dynamic social networks due to a long time-consuming computation. Subsequently, the CELF algorithm proposed by Leskovec et al. [20] and the CELF++ algorithm developed by Goyal et al. [21] aims to improve the scalability and time efficiency by separately using the sub-modularity and marginal diminishing effect of propagation. Nevertheless, they cannot be applied to a dynamic and large-scale environment by high-efficient computation.

Meanwhile, many heuristic algorithms were developed to reduce the greedy-based algorithms' time complexity in IMP. Random heuristic selects the seeds randomly without an approximate guarantee [1]. The Degree Discount Heuristic (DDH) takes advantage of a deterministic degree strategy for seed set selection [24]. Maximum Degree Heuristic (MDH) and High Page Rank Heuristic (HPRH) respectively utilize the value of nodes' degree and page rank to select the top users into a seed set [44]. Gong et al. [27] proposed a discrete particle swarm optimization (DPSO)

to optimize the local search strategy and speed up the seed set selection. Similarly, the ant colony optimization (ACO) was applied in influence maximization by Singh et al. [29]. They rebuilt the principle of pheromones deposited by ants and relevant heuristic information to optimize the local influence. Li et al. [30] developed a differential evolution algorithm based method to obtain the influencers. Although these heuristic algorithms save time and are highly scalable, they decrease the quality of selected seed sets, hardly achieving influence maximization in dynamic and large-scale social networks.

Furthermore, some community-based algorithms have been established to balance scalability and time efficiency in IMP. The CIM algorithm establishes a community structure of candidate seed sets to select the final seed set for influence maximizing spreading [31]. However, the performance of such an algorithm relies on a few parameters without any approximation guarantee. By contrast, the DIN, a parameterless approach, combines the overlapping community structure and the network semantics on users' interest to identify the seed users from the candidates [32]. Nevertheless, the high time complexity of the DIN becomes an obstacle when dealing with large-scale networks.

For the above classical influence maximization algorithms, Peng et al. [45] presented a survey paper from understanding the social influence to analyzing influence maximization algorithms. One of the research challenges mentioned the dynamic evolution of social networks, which corresponds to the above reviews. Meanwhile, Banerjee et al. [44] discussed the above influence maximization algorithms' types in a survey paper. The classification is mainly based on major research challenges concerned with IMP, such as the practicality of realistic networks and the balance between accuracy and computational time. Therefore, most classic influence maximization algorithms are not specifically targeting dynamic social networks, and they present several shortcomings in the balance of effectiveness and efficiency in large-scale networks.

2.2. Dynamic influence maximization

Dynamic influence maximization methods aim to address real-time IMP by capturing constantly evolving network topology and uncertain users' features in dynamic social networks, which are closer to the real-world propagation environment. Bian et al. [47] reviewed the algorithms to identify the top K influence nodes, finding the research trend shifting from computation efficiency and scalability to dynamic networks in current years. Many studies have been conducted to investigate influence maximization in dynamic social networks.

Zhuang et al. [19] propose the Maximum Gap Probing (MaxG) algorithm to approximate the influence maximization by minimizing the possible gap of the probing nodes between the observed network and the actual network. However, even though the real-time performance of the MaxG algorithm appears outstanding, its stable performance is limited by the value of the tolerance probability. Similarly, Han et al. [48] develop a practical dynamic probing framework by utilizing a proposed divide-and-conquer strategy to deal with the natural changes in social networks. Nevertheless, this method only probes several communities to increase the time efficiency [48], which lacks the individual and global view of the network topology. Tong et al. develop an Adaptive Greedy (A-Greedy) algorithm and an adaptive Heuristic Greedy (H-Greedy) based on the Dynamic Independent Cascade (DIC) model to optimize the influence maximization solution [49]. Moreover, Wang et al. propose the Influential Checkpoints (IC) framework and its upgraded version Sparse Influential Checkpoints (SIC) frameworks to handle the continuous dynamic IMP over high-speed social streams [50]. Although these two frameworks have the superiority of optimization efficiency, the final influence maximization results do not outperform other baselines. Furthermore, Murata and Koga developed Dynamic Degree Discount, Dynamic CI, and Dynamic RISA by extending previous static methods to dynamic social networks [51]. However, the performances of these dynamic approaches are not better than the greedy algorithm [51]. Li et al. propose a dynamic algorithm based on cohesive entropy to identify the most influential nodes by considering the overlapping community and optional dynamic influence propagation [52]. However, the experiments did not demonstrate the performance of its real-time influence maximization. Meanwhile, this proposed dynamic algorithm does not demonstrate advent advantages when applied to a large-scale social network.

Additionally, Both Li et al. [46] and Hafiene et al. [53] comprehensively reviewed the influence maximization algorithms and the dynamic IM solutions. The former emphasized the design objective and application of methods in the different social network contexts, discussing the concept, boundary and solutions of the dynamic IMP. The latter subdivided the dynamic networks into snapshot networks and dynamic networks and summarized the limitations of space and time when dynamically searching the optimal influencers.

Therefore, most existing dynamic influence maximization algorithms present a limitation in balancing the effectiveness and efficiency to a different extent.

2.3. GA-based influence maximization

GA, inspired by the "survival of the fittest" theory, has been applied to IMP in recent years. In GA, an individual encoded as a limited size of chromosomes represents a potential seed set, while a gene of the chromosome refers to a seed user. Meanwhile, GA can continuously optimize the potential seed sets by using the operators including selection, crossover and mutating, until it gets close to the optimum solution. Tsai et al. combine GA with the greedy algorithm to improve the effectiveness of the IMP solution [54]. Bucur and Iacca leverage a simple genetic algorithm attaining multiple diverse solutions that show equally high network influence without any assumptions about the network structure [41]. Kromer and Nowakova extend GA by considering the guiding concept to narrow down search space and enhance the evolution efficiency for addressing a fixed-length subset selection in IMP [55]. Zhang et al. adopt multi-populations in GA to ensure the diversity of algorithm solutions and optimize the seed sets through the competition and evolution [23]. Agarwal and Mehta utilize GA with dynamic probabilities that are associated with nodes' out degree in real-time, aiming to find the optimal seed set and reach the maximum influence coverage [56]. Cui et al. propose the Degree-Descending Search Evolution (DDSE) by integrating a degree-descending search strategy with an evolutionary algorithm to overcome the time efficiency issue of greedy-based algorithms [57]. Konotopska and Iacca develop graph-aware evolutionary algorithms to optimize influence maximization outcomes and reduce running time through approximate fitness functions and graph-aware mechanisms [14]. Wang et al. study the IMP in multi-layer networks and propose a multi-factorial evolutionary algorithm with problem-directed operators, where the informative knowledge from both genetic and fitness domains is combined [58]. Even though the above GA-based approaches in IMP demonstrate several advantages such as the optimization process, time efficiency, and the diversity of solutions, most of them cannot handle the dynamics of social networks, especially a mass of individual dynamic information. With an exception, Lotf et al. propose a dynamic generalized genetic algorithm to address the influence maximization problem in a dynamic social network [59]. Whereas, the

proposed method is centralized, requiring the entire dynamic network as the algorithm input. This inevitably leads to the issue of high computational space. In contrast, our approach is decentralized, having the computations of influence capability estimation distributed to the individual users.

Besides, ABM has been widely acknowledged as an appropriate tool for modeling complex systems by defining the problems at a microscopic level. Li et al. leverage ABM to assist in capturing both individual's behaviors and influence status at a microscopic level for modeling influence diffusion [40] and then propose an Enhanced Evolution-Based Backward (2E2B) algorithm [38] for mining influencers during networked evolutionary. Even though this algorithm can capture dynamic network information, the selected seed set is only optimized by measuring the individual agent's influence maximization, lacking the seed set optimization from the global level.

Accordingly, compared with Greedy-based, heuristic-based, community-based algorithms and dynamic influence maximization approaches, GA presents a comprehensive advantage in optimal accuracy and time cost. Meanwhile, combining GA and ABM turns out to be a promising approach for IMP in dynamic social networks. This hybrid model takes the advantage of the optimization process, time efficiency and diverse solutions of GA, which enables the networked dynamic features' acquisition from ABM for addressing IMP in dynamic and large-scale social networks. In the next section, we will elaborate on ABEM and explain to achieve influence maximization in dynamic and large-scale networks.

3. Formal definitions and problem formulation

In this section, we give formal definitions relevant to the ABEM approach and formulate the influence maximization problem in a dynamic environment.

3.1. Formal definitions

We start with the fundamental concepts, including graph, user, neighbors and edges. A graph $G = (V, E)$ is defined as a collection of users $V = \{v_1, v_2, \ldots, v_n\}, n \in \mathbb{N}$ with the corresponding connections $E = \{e_{ij} | i, j \in \mathbb{N}, i \neq j\}$. User v_i has a set of neighbors Γ_{v_i}. The degree of v_i refers to the cardinality of the neighbors, i.e., $|\Gamma_{v_i}|$. If there is a connection between v_i and v_j, we have $e_{ij} \in E$, $v_i \in \Gamma_{v_j}$ and $v_j \in \Gamma_{v_i}$. Edge e_{ij} is represented as a tuple, i.e., $e_{ij} = (v_i, v_j)$, implying a potential influential relationship between v_i and v_j.

Definition 1. A dynamic social network $G_D = \{G(t) | t \in \mathbb{N}\}$ is defined as a sequence of graphs, capturing the graph snapshots over time. Thus, $G(t) = (V(t), E(t))$, where t denotes the time step and $G(t)$ refers to the network state at t. $V(t)$ and $E(t)$ describe the users and edges of the network at t, respectively. The users and edges of network $G(t)$ at t are fixed, but they evolve over time.

Definition 2. A user agent $v_i \in V(t)$ refers to an active, autonomous and interactive user in social network $G(t)$. In the dynamic environment, v_i may or may not exist in $V(t+1)$, and $\Gamma_{v_i}(t)$ also changes at the next time step $t+1$.

User agent v_i is capable of accessing its local context, i.e., all the information about neighbors and the edges. In particular, $\Gamma_{v_i}(t)$ describes the adjacent neighbors of v_i at t. Node degree $d_i(t)$ of v_i denotes the size of v_i's neighborhood $\Gamma_{v_i}(t)$ at t. Mathematically, $d_i(t)$ is represented by using the size of v_i's neighborhood, i.e.,

$$d_i(t) = |\Gamma_{v_i}(t)|, \tag{1}$$

Meanwhile, within a limited local view, user agent v_i conducts influence capability estimation with the assistance of neighborhood, where the influence capability describes the number of users influenced by v_i. Specifically, given a limited number of influence diffusion level l, agent v_i diffuses an influence with a maximum hop of l based on the classic IC model [1,24,60]. Agent v_l, requiring l hops to reach v_i, can provide status feedback to v_{l-1}, and so on and so forth. The count of the influenced users is regarded as the degree of influence capability in this single attempt. Multiple trials are initiated, and the average value is used as the influence capability estimated by v_i, which is denoted as $\sigma(v_i)$.

Definition 3. Influencer pool $C(t)$ is defined as a collection of influencer candidates at t. These candidates' influence capabilities are over a pre-defined threshold θ_s and greater than θ_q percentage of their neighbors, i.e.,

$$C(t) = \{v_i | v_i \in V(t) \wedge \sigma(v_i) > \theta_s \wedge \frac{|\sigma(v_i)'|}{|\Gamma_{v_i}|} > \theta_q\}, \tag{2}$$

where

$$|\sigma(v_i)'| = |\{v_j | v_j \in \Gamma_{v_i} \wedge \sigma(v_i) > \sigma(v_j)\}| \tag{3}$$

Users from the influencer pool are potentially selected as members of the seed set (refer to Definition 4). $C(t)$ is constructed through proactive proposals initiated by user agents $V(t)$.

The influencer pool is shared by all the user agents. The size of the influencer pool varies according to the changing network topological structure, e.g., $|C(t)|$ can be different from $|C(t+1)|$. An element $v_c \in C(t)$ describes an influencer candidate. Specifically, user agent v_c proactively estimates the influence capabilities against Γ_{v_c} at t and determines whether to propose as one of the potential influencers in $C(t)$. The detailed behavior is described in Algorithm 1. In the current context, user agent v_c intends to nominate itself as a seed candidate only when its influence capability exceeds a particular threshold θ_s and is greater than θ_q percentage of their neighbors at t.

Definition 4. A seed set $S(t) = \{v_1, v_2, \ldots, v_k\}, S(t) \subseteq V(t)$ refers to a finite set of identified influencers from the social network $G(t)$ at t, where $k = |S(t)|$ represents the number of influencers needs to be selected. The influencers selection algorithm is named as seeding algorithm.

In GA, a seed set corresponds to a *"chromosome"* or an *"individual"*. Mapping to the problem space, each chromosome or individual implies a potential solution to the problem. Within a seed set, each element is called a *"gene"*.

Definition 5. A population generally refers to a collection of candidate solutions for a pre-defined problem space in GA. In the current setting, a population $R_i(t)$ corresponds to a collection of seed sets for $G(t)$, which are recognized as the potential solutions to the influence maximization problem. Specifically, population $R_i(t) = \{S_1(t), S_2(t), \ldots, S_j(t)\}$ represents the $i-$th generation of the overall evolution process, and $S_j(t)$ represents a candidate solution of the seed set. R_0 means the initial generation. $R_i(t)$ evolves to the next generation $R_{i+1}(t)$ through the GA operators.

3.2. Problem description

Given a dynamic social network $G(t) = (V(t), E(t))$ and an integer k at t, the objective is to efficiently select k users from $V(t)$ as the seed set $S(t)$, expecting they can spread influence and maximize the impact $\sigma(S(t))$ across $G(t)$. $\sigma(S(t))$ represents the influence coverage, describing the expected number of influenced

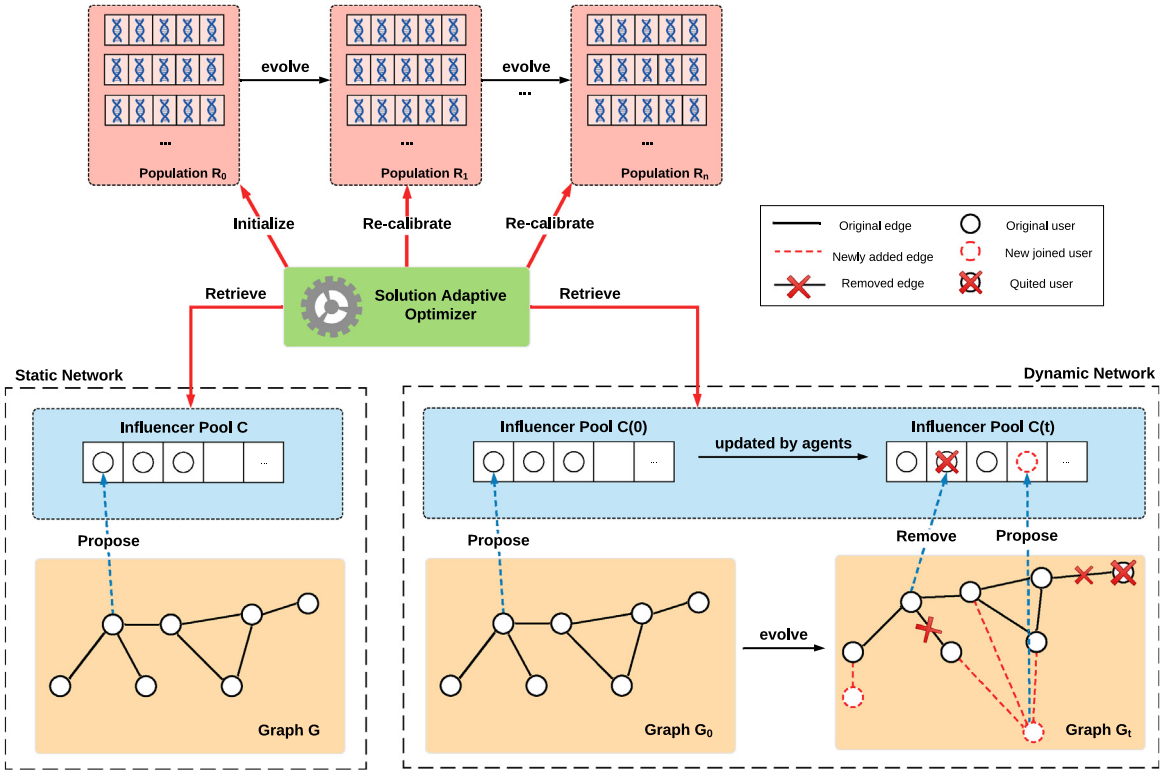


Fig. 1. Overall process of ABEM approach.

users at the end of the diffusion process if $S(t)$ is selected as a seed set. Note that our target is to maximize $\sigma(S(t))$ at every time step, as the dynamic social network keeps evolving over time. The objective is described in Eq. (4).

$$\max \sum_{t} \sigma(S(t)) \tag{4}$$

After n time steps, given $G(t+n) = (V(t+n), E(t+n))$, the originally identified seed set $S(t)$ needs to be adapted efficiently as $S(t+n)$ to fit the new problem space without re-calculation. Overall, the solution requires to be adapted automatically with the rapid evolution of the online social network.

4. Agent-based evolutionary model for mining influencers

In this section, we first explain the overall process of mining influencers from social networks by leveraging ABEM. Next, we drill down to the details from both macro-perspective, i.e., user agent's behaviors modeling, and micro-perspective, i.e., adaptive solution optimization, where the algorithms will be elaborated as well.

4.1. Overall process of ABEM

ABEM facilities the advantages of both agent-based modeling and evolutionary computation, where both agent's local computational power and a centralized optimizer are adopted. Fig. 1 illustrates the main idea of ABEM.

The key process starts from local user agents from Graph $G(0)$, which proactively evaluate its influence capabilities by comparing against the neighbors and decide whether to propose themselves as influencers via merging into the influencer pool $C(0)$. At the time step t, if the local environment of user agent v_i changes, e.g., establishing new links with others, v_i will re-assess the influence capabilities and update the influencer pool $C(t)$ at t. In this way, the influencer pool, shared by all the agents, always can be kept up-to-date. More importantly, the influencer pool narrows down the search scope of a large-scale network, leaving only a small amount of data for a centralized component to process.

The adaptive solution optimizer plays a pivotal role in ABEM, which is capable of retrieving influencer candidates from the influencer pool in real-time. Meanwhile, it optimizes the solutions generated by GA by re-calibrating the "genes". With the evolution of GA, the adaptive solution optimizer contributes to the modification of the population, leading each generation to rapidly evolve toward reaching optimal solutions. The optimal solution is a seed set that has the maximum influence coverage. The influence coverage $\sigma(S_j(t))$ refers to the influence capability of the seed set $S_j(t)$. $\sigma(S_j(t))$ can be presented as:

$$\sigma(S_j(t)) = \sigma(\{v_i | v_i \in S_j(t)\}) \tag{5}$$

In a nutshell, each user agent undertakes a self-evaluation of the corresponding influence capability, assisting in identifying the potential influencers. This can effectively handle the dynamics of large-scale social networks. Meanwhile, the evolutionary algorithm drives the seed selection process by continuously optimizing the solutions. Next, we elaborate on the modeling of user agents' behaviors and adaptive solution optimization in detail. The notations used in the following algorithms are listed in Table 1.

4.2. Influencer nomination by autonomous agents

In ABEM, users are modeled as autonomous and proactive agents, which are capable of communicating with their neighbors, retrieving information from the local environment and estimating the influence capability. Influencer nomination is initiated by user agents proactively. The outcome of such nomination is revising the influencer pool, keeping it update-to-date with evolving social networks.

Table 1
Notations description.

Notations	Description
p_a	Activation probability of diffusion model
$R_0(t)$	The initial/first population for $G(t)$
$\|R_0(t)\|$	The size of $R_0(t)$
$S(t)$	The optimal chromosome/seed set/solution for $G(t)$
k	The size of chromosome $S(t)$, $k = \|S(t)\|$
p_s	Selection rate
p_c	Crossover rate
p_m	Mutation rate
p_d	Degree change rate

Algorithm 1 describes the agent-based influencer nomination process, where all the computations are conducted by agents locally. The inputs include user agent v_i, the degree threshold θ_s, and influence quartile threshold θ_q. The output is the updated influencer pool $C(t)$ at time step t. Lines 1–3 aim to initialize variables and examine the neighborhood at t. Lines 4–9 request neighbors' update-to-date information, including the influence capability, and calculate the count of neighbors whose influence capability is weaker than v_i. Lines 10–14 compare the influence capability and influence quartile of v_i against the thresholds θ_s and θ_q, respectively, and determine how to update the influencer pool.

Algorithm 1 Agent-based Influencer Nomination Algorithm.

Input: User agent v_i, degree threshold θ_s, and influence quartile threshold θ_q
Output: Decision of nomination by updating influencer pool $C(t)$

1: Obtain the up-to-date neighbors Γ_{v_i}
2: Estimate v_i's degree $|\Gamma_{v_i}|$ and influence capability $\sigma(v_i)$
3: Initialize $var := 0$
4: **for** $v_j \in \Gamma_{v_i}$ **do**
5: Request v_j's influence capability $\sigma(v_j)$
6: **if** $\sigma(v_i) > \sigma(v_j)$ **then**
7: $var := var + 1$
8: **end if**
9: **end for**
10: **if** $(\frac{var}{|\Gamma_{v_i}|} > \theta_q) \wedge (\sigma_{v_i} \geq \theta_s)$ **then**
11: $C(t) := \{v_i\} \cup C(t)$
12: **else if** $v_i \in C(t)$ **then**
13: $C(t) := C(t) \setminus \{v_i\}$
14: **end if**

4.3. Adaptive solution optimization

Adaptive solution optimization incorporates two key concurrent processes, i.e., solution optimization and adaptation. In the former, GA has been adopted to continuously optimize the solutions over time, where the search space is defined by the Adaptive Solution Optimizer (ASO). In the latter, ASO re-calibrates the solutions by considering both the update-to-date influence pool and the existing outcome. Obviously, ASO plays a critical role in bridging the decisions from user agents and the outputs from the evolutionary algorithm, which also reflects the main idea of our proposed ABEM for addressing the influence maximization problem.

Algorithm 2 describes the overall process of mining influencers by facilitating ABEM, where four fundamental operators, including initialization, selection, crossover, and mutation, are tailored to fit the problem space. Furthermore, since the evolution of networks can be reflected by the variation of the influencer pool, the solutions will be adapted between two consecutive generations by replacing the outdated candidates with new influencers who appeared in the influencer pool, namely, a re-calibration operation.

Algorithm 2 ABEM for Influence Maximization.

Input: Dynamic social network $G(t)$, influencer pool $C(t)$, activation probability p_a, crossover rate p_c, mutation rate p_m, and seed set size k
Output: A seed set $S(t) = \{v_1, v_2, \ldots, v_k\}$.

1: Initialize generation $i = 0$, Population $R_i(t)$, a seed set $S(t) = \emptyset$
2: Evaluate influence coverage $\forall S_j(t) \in R_0(t)$, calculate $\sigma(S_j(t))$ using Eq. (5)
3: Find out the best solution $S_{fittest}(t)$ from $R_0(t)$ and assign to $S(t)$
4: **while** !Termination Condition **do**
5: Start re-calibration $Rec[C(t), R_{i-1}(t)] \rightarrow R_i(t)$
6: Start selection $Sel[R_i(t)] \rightarrow R_i(t)$
7: **if** random $\xi_c < p_c$ **then**
8: Start crossover $Cro[R_i(t), C(t), p_c] \rightarrow R_i(t)$
9: **end if**
10: **if** random $\xi_c < p_m$ **then**
11: Start mutation $Mut[R_i(t), C(t), p_m] \rightarrow R_i(t)$
12: **end if**
13: Evaluate influence coverage $\forall S_j(t) \in R_i(t)$, calculate $\sigma(S_j(t))$ using Eq. (5)
14: Find the solution $S_{fittest}(t)$ with the greatest influence coverage
15: **if** $\sigma(S_{fittest}(t)) > \sigma(S_j(t))$ **then**
16: $S(t) := S_{fittest}(t)$
17: **end if**
18: $i := i + 1$.
19: **end while**
20: **return** The seed set $S(t)$

Lines 1–3 initialize a population from the current influencer pool $C(t)$ and evaluate the influence coverage of each individual, where the output of fitness function $\sigma(\cdot)$ represents the estimated influence coverage based on the IC model. Line 4 starts the seeding process, where the termination condition is triggered when $\sigma(\cdot)$ of the best solution S_m starts to converge and remains unchanged for a number of generations. This is measured within a fixed number g of generations, i.e., the maximum number of iterations. Lines 5–12 run through the operators for optimization, where $Sel(\cdot)$, $Cro(\cdot)$, $Mut(\cdot)$ and $Rec(\cdot)$ represent the selection, crossover, mutation and re-calibration operators, respectively. Lines 13–18 evaluate the fitness value of each solution yield by the current generation and find out the best solution, i.e., a seed set with the greatest influence coverage. Line 20 returns the best solution.

In the following subsection, we explain the key operators utilized in ABEM, including initialization, selection, crossover, and mutation, as well as re-calibration for solution adaptation.

4.4. Key operators

Population Initialization. The initial population of seed sets turns out to be very important since it defines the starting point of exploring the "best" solution, i.e., a seed set, for the influence maximization problem. ASO generates the initial population $R_0 = \{S_1, S_2, \ldots, S_m\}$ with a population size of $|R_0| = m$, i.e., a collection of m candidate solutions (*chromosomes* or *seed sets*). A chromosome $S_m = \{v_1, v_2, \ldots, v_k\}$ corresponds to one seed set, where $k = |S_m|$ denotes seed set size.

Selection Operator. The selection operator assists in identifying a collection of seed sets for further improvement, where the

fitness value of each individual is considered the key factor. In other words, seed sets with a higher influence coverage have a greater chance to be selected for the next generation. Moreover, since both original solutions and modified ones via the operators will remain, the number of candidate solutions is much greater than the population size. The selection operator also filters out the "bad ones" and controls population size.

Therefore, the selection rate p_s of an individual $S_j(t)$ from population $R_i(t)$ to be selected can be formulated in Eq. (6). $N = |R_i(t)|$ is the population size of $R_i(t)$.

$$p_s[S_j(t)] = \frac{\sigma(S_j(t))}{\sum_{j=1}^{N} \sigma(S_j(t))}. \tag{6}$$

Algorithm 3 Selection Operator.

Input: An augmented population $R_i(t)$, target population size $|R'_i(t)|$.
Output: Population $R'_i(t)$ after selection.

1: **if** $|R_i(t)| == |R'_i(t)|$ **then**
2: return $R_i(t)$
3: **end if**
4: Initialize $R'_i(t) := \emptyset$
5: **for** $\forall S_j(t) \in R_i(t)$ **do**
6: Estimate the influence coverage of $S_j(t)$, i.e., $\sigma(S_j(t))$ using Eq. (5)
7: Calculate selection rate $p_s[S_j(t)]$ using Eq. (6)
8: **if** random $\xi_s < p_s[S_j(t)]$ **then**
9: $R'_i(t) := R'_i(t) \cup \{S_j(t)\}$
10: **end if**
11: **end for**
12: **while** $|R'_i(t)| < |R_i(t)|$ **do**
13: Select $S_x(t), \sigma(S_j(t)) \geq \forall S_j(t) \in R_i(t) \wedge S_x(t) \notin R'_i(t)$
14: $R'_i(t) := R'_i(t) \cup \{S_x(t)\}$
15: **end while**
16: **Return** $R'_i(t)$.

Algorithm 3 describes how the selection operator works. Lines 1–3 check the size variance and determine if the selection continues or not. Lines 4–11 copy over the solutions from an augmented population $R_i(t)$ to $R'_i(t)$ based on the selection probability in Eq. (6). Lines 12–15 fill $R'_i(t)$ if the size of $R'_i(t)$ does not reach $|R_i(t)|$.

Crossover Operator. The crossover operation in the influence maximization problem recombines two seed sets (parents) and generates two new solutions (offspring). In other words, two selected seed sets exchange the influencers at a random slicing point and produce two new seed sets. Mixing two solutions may cause duplicated elements in a seed set. Thus, a repair function is required to fix the solution by replacing the duplicated influencer with a random user from the influencer pool.

The crossover operator is described in Algorithm 4. Lines 2–3 check if the current seed set S_m is selected for crossover. Lines 5–7 prepare the operation by obtaining another seed set, generating a slicing point and initializing two offspring. Lines 8–15 conduct crossover. Lines 16–23 repair the generated seed set by adding users from the influencer pool. Because offspring is modeled as a hash set where duplicated items remain a single copy. In other words, the offspring with a lower size requires to be "fixed". Line 24 expands the current generation by appending the newly generated offspring and Line 26 returns the updated R_i after crossover as R'_i.

Mutation Operator. The mutation operator works on an individual user of a seed set, replacing a specific seed (user) with another potential influencer. This operator helps to maintain the diversity of seed sets from one generation to the next, which enables ABEM to have a wide range of feasible solutions, avoiding rapid coverage to a local optimal solution. Specifically, a seed is supposed to be substituted by a randomly selected seed candidate from the current influencer pool with a certain probability. The operator is described in Algorithm 5.

Algorithm 4 Crossover Operator.

Input: Population $R_i(t)$, influencer pool $C(t)$, and crossover rate p_c.
Output: Population $R'_i(t)$ after crossover.

1: **for** $\forall S_j(t) \in R_i(t)$ **do**
2: **if** random $\xi_c > p_c$ **then**
3: next
4: **end if**
5: Select a seed set $S_{fittest}(t) \in R_i(t) \setminus \{S_j(t)\}$ which is the best solution in $R_i(t)$
6: Randomly select a slicing point ξ_s
7: Initialize two offspring solutions, i.e., offspring_1, offspring_2
8: **for** i in range(0,ξ_s) **do**
9: offspring_1.add($S_j(t)$.get(i))
10: offspring_2.add($S_{fittest}(t)$.get(i))
11: **end for**
12: **for** i in range(ξ_s, $|S_j(t)|$) **do**
13: offspring_1.add($S_{fittest}(t)$.get(i))
14: offspring_2.add($S_j(t)$.get(i))
15: **end for**
16: **while** $|\text{offspring}_1| < |S_j(t)|$ **do**
17: newgene := $C(t)$.get(random($|C(t)|$))
18: offspring_1.add(newgene)
19: **end while**
20: **while** $|\text{offspring}_2| < |S_j(t)|$ **do**
21: newgene := $C(t)$.get(random($|C(t)|$))
22: offspring_2.add(newgene)
23: **end while**
24: $R'_i(t) := R_i(t) \cup \{\text{offspring}_1, \text{offspring}_2\}$
25: **end for**
26: **return** $R'_i(t)$.

Algorithm 5 Mutation Operator.

Input: A population $R_i(t)$, influencer pool $C(t)$, and mutation rate p_m
Output: A mutated population $R'_i(t)$.

1: **for** $S_j(t) \in R_i(t)$ **do**
2: **for** $v_i \in S_j(t)$ **do**
3: **if** random $\xi_m < p_m$ **then**
4: Randomly select $v_k \in C(t) \wedge v_k \notin S_j(t)$.
5: $v_k \rightarrow v_i$
6: **end if**
7: **end for**
8: **end for**
9: **return** $R'_i(t)$.

Re-calibration Operator. The re-calibration operator aims to adapt the existing population based on the changing environment. As the influence capabilities of the seeds introduced to the current population vary over time, it is essential to update the existing solution by replacing partial "out-dated" influencers. Specifically, the re-calibration operator checks through all the seed sets within a population, figuring out the users whose influence capabilities are degraded significantly. Such users will be replaced with those who are newly introduced to the influence pool.

Algorithm 6 describes the re-calibration process. The inputs include user set $V(t)$ at time step t, influencer pool $C(t)$, and the current population $R_i(t)$ to be adapted. The output is re-calibrated population $R_i'(t)$. Lines 3–5 identify the users from $R_i(t)$, who quit the network and replace these users with a randomly selected user from the influence pool. Lines 6–13 initiate adaptation based on the estimated degree variation rate.

Algorithm 6 Re-calibration operator.

Input: A user set $V(t)$, influencer pool $C(t)$, current population $R_i(t)$
Output: Re-calibrated population $R_i'(t)$

1: **for** $\forall S_j(t) \in R_i(t)$ **do**
2: **for** $\forall v_i \in S_j(t)$ **do**
3: **if** $\nexists v_i \wedge v_i \in V(t)$ **then**
4: Randomly select $v_k \in C(t) \wedge v_k \notin S_j(t)$
5: $v_k \rightarrow v_i$
6: **else if** $v_i \notin C(t)$ **then**
7: Calculate degree change rate $p_d = 1 - \frac{d_i(t)}{|d_i(t-1)|}$
8: **if** random $\xi_d < p_d$ **then**
9: Randomly select $v_k \in C(t) \wedge v_k \notin S_j(t)$
10: $v_k \rightarrow v_i$
11: **end if**
12: **end if**
13: **end for**
14: **end for**
15: **Return** $R_i'(t)$.

5. Experiments

In this section, four experiments are conducted to evaluate the performance of ABEM. The first experiment analyzes the convergence of ABEM with different experimental settings. In the second experiment, we evaluate the continuous performance of ABEM on influence maximization within a dynamic environment. The third experiment compares the performance of ABEM against several baselines by influence maximization. The last experiment further explores the parameter settings of ABEM. The following subsections introduce the experimental settings, present the experiment details and discuss the results, respectively.

5.1. Experimental settings

Three real-world datasets are utilized for the experiments, including Ego-Facebook,[1] [61] Git,[2] [62] and Flixster.[3] [63] The properties of these datasets are described in Table 2, and the parameters of ABEM are listed in Table 3.

Influence coverage, i.e., the classic evaluation metrics of the influence maximization problem, is adopted for all the experiments. Influence coverage refers to the number of users activated (influenced) by the identified influencers. On top of that, we use elapsed running time as evaluation metrics in Experiment 3, indicating the time cost for the algorithm to find the solutions. To conclude, influence coverage and running time represent the effectiveness and efficiency of the proposed algorithm, respectively.

The following baselines for the influence maximization problem are utilized for performance comparison, where the greedy algorithm is recognized as one of the strongest baselines.

1 https://snap.stanford.edu/data/ego-Facebook.html
2 http://snap.stanford.edu/data/github-social.html
3 http://www.flixster.com/

Table 2
Datasets.

Network	No. of nodes	No. of edges	Type
Ego-Facebook	1,899	20,296	static, undirected
Git	13,419	59,259	static, undirected
Flixster	14,231	79,265	dynamic, directed

Table 3
Parameters of ABEM.

Parameter	Description	Value
p_c	Crossover rate	1
p_m	Mutation rate	0.1
$\|R_i(t)\|$	Population size, the number of chromosomes in one population $R_i(t)$.	50
g	Generation numbers, the maximum number of iterations	1000

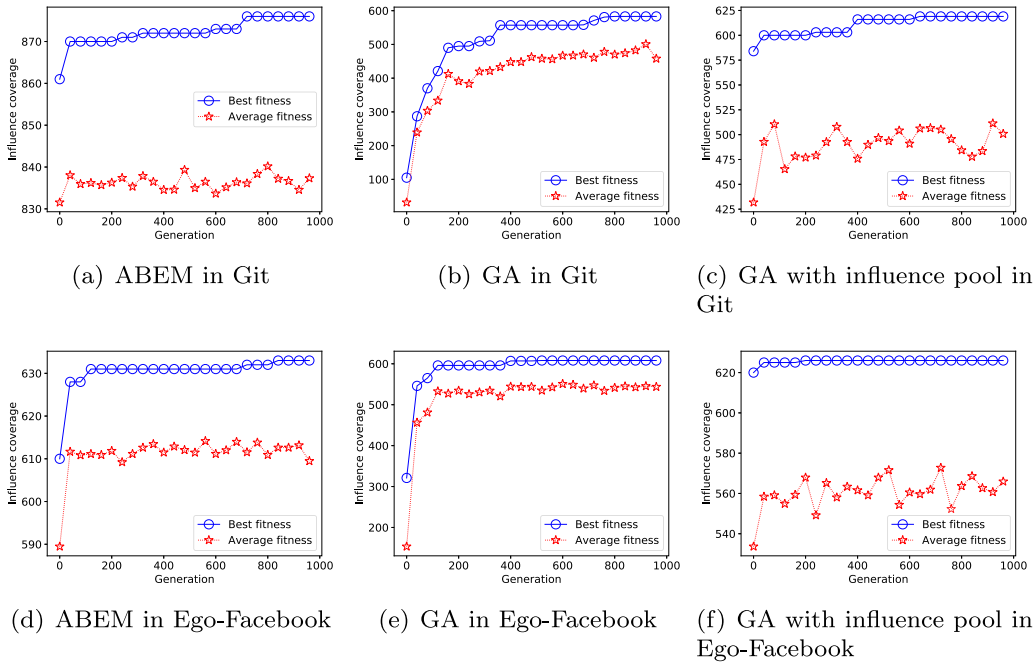
- Greedy: Each seed is selected by iterating all the users, aiming at reaching the maximum influence marginal gain. The greedy algorithm is not scalable as it relies on enormous times of Monte-Carlo simulations.
- Degree-based selection: Users with the highest degree will be selected as influencers.
- Degree Discount Heuristics (DDH): The seeds are selected by deterministic degree strategy. This algorithm is developed based on the idea that users with high degrees normally cluster together. Hence, it is not necessary to select all of them [24].
- Genetic Algorithm (GA): The traditional GA without any optimization or tailoring. Specifically, the seeds are selected after evolving for a few generations using classic GA operators.
- GA with influence pool: The traditional GA with an optimized initial population, where the solutions are initialized based on the influencer pool.
- Random: The seeds are randomly selected at each time step. The executing time is the fastest but normally with the lowest influence coverage as it does not follow any heuristics.

5.2. Experiment 1: Convergence analysis

Experiment 1 analyzes the convergence of ABEM by tracking the evolution pattern of the continuously optimized solutions, i.e., seed sets. For each generation, both average influence coverage (average fitness) and the highest influence coverage (the best fitness) are estimated. The evolutionary algorithms are validated by using two datasets, i.e., Git and Ego-Facebook. The experiment also defines a fixed number of generations, i.e., 1000.

Fig. 2 shows the evolutionary trends for each population by facilitating ABEM, GA and GA with influencer pool. It is evident that after a series of generations, all the three algorithms start to converge, and finally reach an optimal solution. Furthermore, ABEM demonstrates the best performance of all, which can be revealed by comparing the influence coverage of the best solution against others at the end of 1000 generations.

The experimental results also implicitly reveal the advantages of ABEM. First, ABEM converges much faster than that of GA, which can be observed by comparing Fig. 2(a) with Fig. 2(b), or Fig. 2(d) with Fig. 2(e). The reason is ABEM leverages the influencer pool for initialization, which enables ABEM to start with a higher point and have a better chance to obtain an optimal solution at speed. Second, ABEM still has a greater chance to improve the existing solutions even reaching a convergence status. Whereas, GA with an influencer pool almost makes no chance



(a) ABEM in Git (b) GA in Git (c) GA with influence pool in Git

(d) ABEM in Ego-Facebook (e) GA in Ego-Facebook (f) GA with influence pool in Ego-Facebook

Fig. 2. Convergence analysis.

Table 4
Flixster quarterly network.

Snapshot ID	Quarter	No. of nodes	No. of links	Average degree
1	2006 Q1	463	1050	4.54
2	2006 Q2	564	1614	5.72
3	2006 Q3	825	2146	5.2
4	2006 Q4	1287	4027	6.26
5	2007 Q1	2958	8965	6.06
6	2007 Q2	2993	8611	5.75
7	2007 Q3	2051	5789	5.65
8	2007 Q4	1280	3517	5.5
9	2008 Q1	1387	4252	6.13
10	2008 Q2	1529	4758	6.22
11	2008 Q3	1615	4610	5.71
12	2008 Q4	1353	3907	5.78

after convergence. This is because the ABEM clearly defines the scope of searching influencers, but the other algorithms conduct the search process in the world. Therefore, in ABEM, the average fitness of all the populations always appears higher than those of the others. Third, ABEM demonstrates greater computational efficiency. Based on the oscillation degree of average fitness, ABEM shows a relatively stable trend. Whereas, the average fitness of other algorithms fluctuates significantly. The reason behind this also relies on the search scope. In ABEM, the influencer exploration scope is narrowed down by the individual user agents, which greatly reduces the centralized computational cost. By contrast, the other algorithms have to handle a larger scope with dramatic changes in the population.

5.3. Experiment 2: Continuously influence maximization in dynamic social networks

Experiment 2 aims to evaluate the continuous performance of ABEM on influence maximization within a dynamic environment. This experiment also explicitly demonstrates the adaptability of ABEM, namely, continuously updating the identified seed sets in a changing environment and adapting the solutions based on past experiences.

In this experiment, the dynamic environment is simulated by using 12 consecutive quarters' transactions from the Flixster dataset, ranging from 2006 to 2008 [63]. The statistics of the dataset are listed in Table 4. Since the size of some snapshots appears small, we give $k = 5$, $\theta_s = 2$, and $\theta_q = 0.7$. Five seeding algorithms, i.e., Greedy, Degree, DDH, GA, and GA with influence pool, are utilized as the counterparts. In the influence maximization problem, the greedy algorithm is recognized as one of the strongest baselines [1,24]. On top of that, the IC model is adopted as the influence diffusion model, with a uniform probability of 0.1, and a number of Monte-Carlo simulations of 100. We also list the assumptions as follows.

- A user joins the network when giving the first rating, and quits after the last rating. A user only can be influenced when he or she appears as an active user in the social network.
- When a user joins the network, the corresponding relationships are established immediately. Likewise, the associated links are removed if the user quits the network.
- The greedy algorithm recalibrates the selected seed set on an annual basis. This is because the greedy algorithm is not scalable for large-scale networks. It is unrealistic to launch the greedy algorithm frequently.
- The Degree and DDH reselect influencers every four quarters as well. This is because these heuristic algorithms require the entire network topology. It is unrealistic to rank all of the users' degrees in large-scale networks frequently.

First, we compare the influence coverage [1,64] of ABEM against the other baselines with different seed set sizes, i.e., $k = 5$, $k = 10$, and $k = 15$. Fig. 3 illustrates the experimental results using 12 consecutive quarters, where the network topology, including both nodes and links, evolves over time. Figs. 3(a) to 3(c) compare the influence coverage at each quarter. Figs. 3(d) to 3(f) compare the accumulative influence coverage, i.e., the overall influence coverage from time step 0 to t, at each quarter. As can be seen from the figure that ABEM outperforms the classic seeding algorithms, implying that the greedy algorithm losses the advantages in a changing environment without any calibration. By contrast,

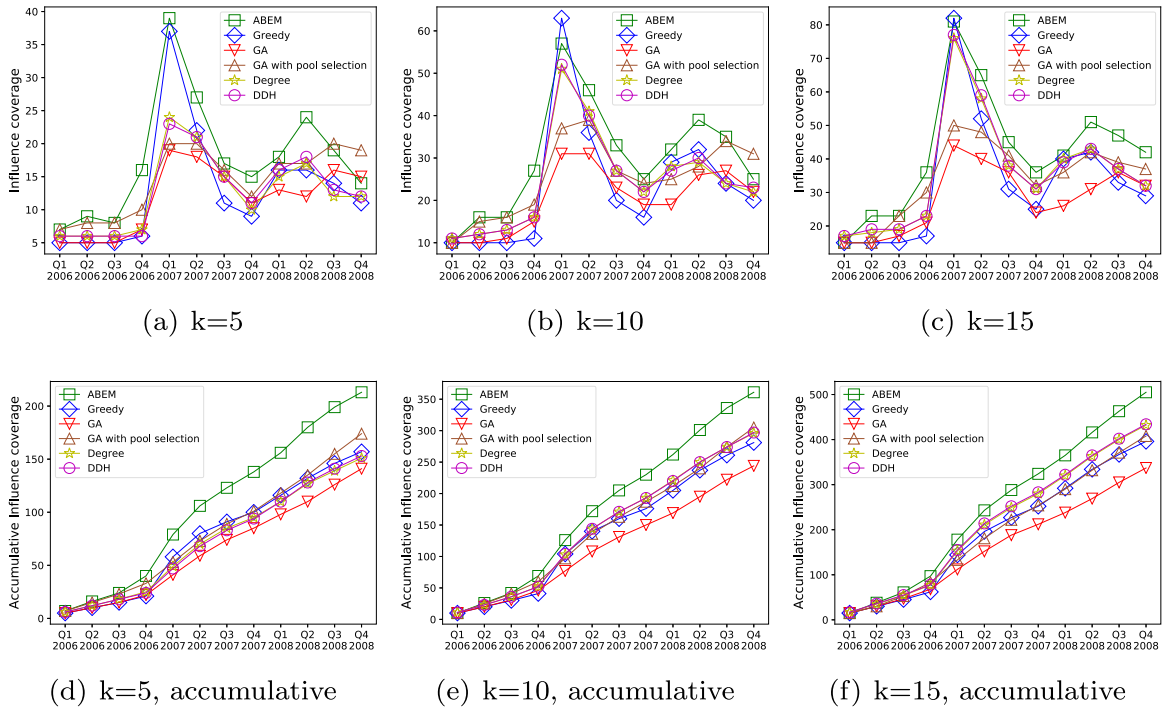


Fig. 3. ABEM performance in Flixster quarterly network.

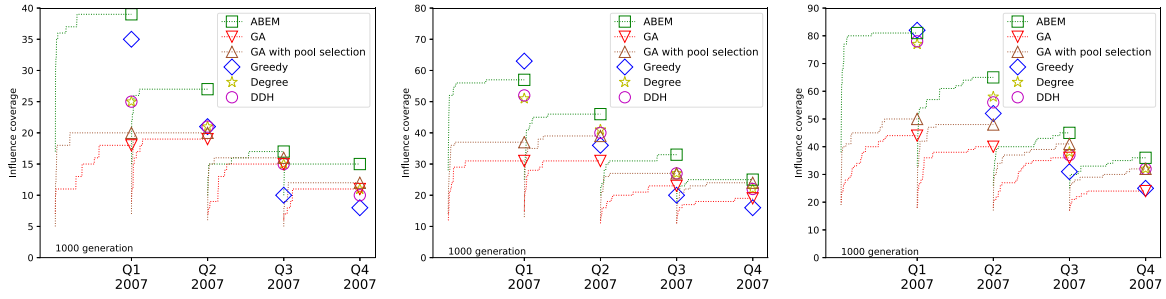


Fig. 4. ABEM performance in dynamic environment.

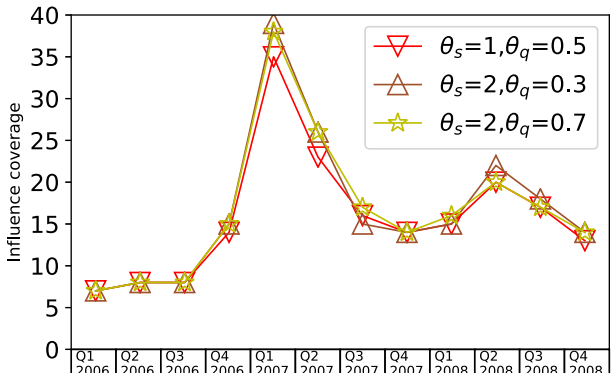
ABEM adapts the solutions over time, and this feature stems from its internal design, i.e., a hybrid of multi-agent systems and evolutionary computation. Furthermore, ABEM outperforms the other two evolutionary approaches, i.e., GA and GA with influence pool algorithms, in dynamic networks. This is because ABEM leverages the influencer pool for initialization and re-calibration, which gives ABEM a better starting point and a higher chance to fast converge to an optimal solution.

Second, we further investigate how ABEM adapts the seed set in dynamic networks by using the same dataset, where four consecutive quarters from Flixster, i.e., from 2007 Q1 to 2007 Q4, are selected. The evolutionary process of ABEM is demonstrated in Fig. 4, where the x-axis represents discrete time steps, i.e., network snapshots, and the y-axis denotes the influence coverage produced by the algorithm. 1000 units are allocated between any two consecutive quarters, and each unit presents a generation produced by evolutionary algorithms. We have selected a seed set size of $k = 5$, $k = 10$, and $k = 15$ for the exploration. As can be seen from these figures that in Q2 2007, ABEM quickly reaches an optimal solution merely within 500 generations. When the network evolves, in Q2 2007, the performance of ABEM drops but climbs up quickly only after a few generations. This is because the existing potential influencers are retained in the influencer pool and the solution can be adapted rapidly. In Q3 2007, ABEM requires a greater number of generations to converge. The reason is dramatic variations occur in the network at this point, many existing influencers need to be replaced within the population. Therefore, it can be proved that ABEM has great adaptability to handle the dynamics of social networks efficiently.

Third, we further validate the adaptive capabilities of ABEM with different parameter settings. Recall that the influencer pool of ABEM is shared by all the agents and scopes the problem search space. Therefore, the influence pool significantly affects the performance of ABEM. In this experiment, we observe the outcome by varying the degree threshold θ_s and influence quartile threshold θ_q of the influencer pool.

Fig. 5 shows the influence coverage with three different settings over 12 quarters of the Flixster dataset: (1) $\theta_s = 1$ and $\theta_q = 0.5$ (2) $\theta_s = 2$ and $\theta_q = 0.3$ (3) $\theta_s = 2$ and $\theta_q = 0.7$, with a seed set size of 5. The table of Fig. 5 depicts the detailed outcome based on various settings.

Not much difference can be observed when the network size is small, i.e., from Q1 2006 to Q3 2006. However, with the increase of network scale, the pool size shows a great impact on influence coverage. A larger pool size ($\theta_s = 1$ and $\theta_q = 0.5$) leads to



	Q1 2006	Q2 2006	Q3 2006	Q4 2006	Q1 2007	Q2 2007	Q3 2007	Q4 2007	Q1 2008	Q2 2008	Q3 2008	Q4 2008
θ_s=1,θ_q=0.5	11	52	103	316	1032	1390	1224	895	1067	1319	1335	1220
θ_s=2,θ_q=0.3	7	28	67	235	924	1125	876	517	728	899	938	812
θ_s=2,θ_q=0.7	5	21	36	125	488	635	529	346	439	580	618	550

Fig. 5. ABEM performance with different pool size.

relatively lower performance. This is because the search space of ABEM is expanded and it requires more generations to reach an optimal solution. By contrast, given a small pool size (i.e., $\theta_s = 2$ and $\theta_q = 0.7$), ABEM yields better performance than those with different settings in most of the snapshots but demonstrates a relatively weak performance in 2008 Q2 and Q3. This suggests that the over shrink of the influencer pool inevitably filters out some potential influencers.

5.4. Experiment 3: Influence maximization comparison

Experiment 3 aims to evaluate the performance of ABEM by classical influence maximization comparison. We compare ABEM against the baselines introduced in Section 5.1, where two static networks, i.e., Ego-Facebook and Git, are adopted. In this experiment, the seed set size k ranges from 5 to 10 with a step of 5. On top of that, it facilitates the IC model as the influence diffusion model, with a uniform probability of 0.1.

The experimental results on Ego-Facebook and Git are demonstrated in Figs. 6(a) and 6(b), respectively. As aforementioned, greedy selection has been recognized as one of the strongest baselines in the influence maximization problem, but not scalable. As can be seen from both figures the greedy selection yields the best performance of these two datasets. Despite carrying out a similar performance as greedy, ABEM is capable of mitigating the scalability issue since the major computations are distributed to the individual agents and the search scope is limited.

In Fig. 6(a), no significant performance difference can be observed among ABEM, Degree, DDH, and Greedy selection. This is because the size of the Ego-Facebook network is small, and the identified seed sets are also similar. Despite this, ABEM performs slightly better than degree and DDH. Given a relatively larger network, in Fig. 6(b), ABEM outperforms other baselines and demonstrates a similar performance as the greedy selection. Notably, when a small seed set is required, e.g., $k = \{5, 10, 15\}$, the performance of ABEM can exceed the greedy selection.

5.5. Experiment 4: Parameter analysis

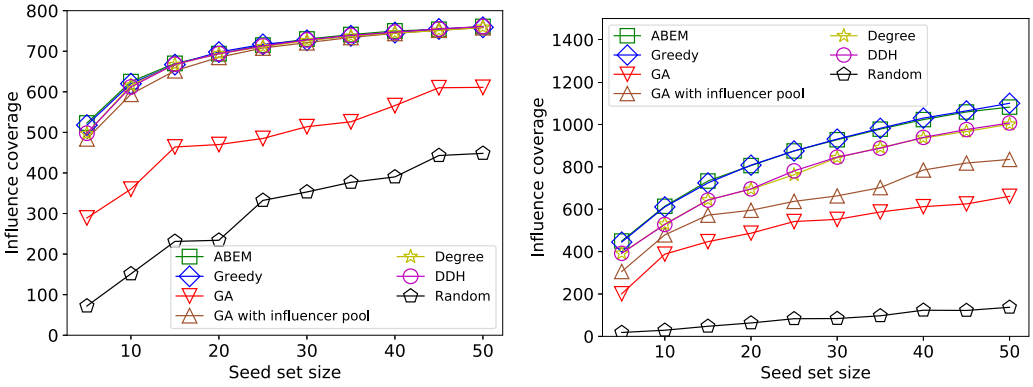
In Experiment 4, we further investigate the performance of ABEM by varying the parameters, including the number of generation g, the degree threshold θ_s, and the influence quartile threshold θ_q.

First, we analyze the impact on the influence coverage and running time by increasing the number of generations. As we can observe from Fig. 7 that with the rise of evolutionary generation, the elapsed running time increases linearly. The influence coverage also demonstrates a steady escalation trend, with slight improvement after 500 generations. This is due to the fact that almost all the potential influencers are incorporated into the population, and it takes time to figure out a better seed set by re-organizing the existing influencers.

Second, we investigate the influencer pool size variations by adjusting the degree threshold θ_s and influence quartile threshold θ_q. Both parameters control the individual's "propose as an influencer" behavior, which directly impacts the size of the influencer pool. Subsequently, it determines the search scope and influences the performance of ABEM. A high degree threshold θ_s implies only those with large neighbor sizes can be proposed as an influencer. Likewise, a high quartile threshold θ_q allows the users who are influencers in their social circle to join the influencer pool.

Figs. 8(a) and 8(b) demonstrate the impact on influencer pool size by varying both parameters in the Ego-Facebook and Git datasets, respectively. It is evident that in both figures, the influencer pool size shows a downside trend with the increase of θ_s or θ_q. However, in the Git dataset, the influencer pool size is more sensitive to θ_s than that of the Ego-Facebook dataset. This is because the node connectivity in Git appears to turn out to be much sparser than Ego-Facebook.

Third, we investigate how the influencer pool size impacts the ABEM's performance. It is important to strike a balance between efficiency and effectiveness. Specifically, a larger influencer pool size enables ABEM to find out a better solution, but with less efficiency due to the large scope. ABEM can converge more rapidly with a smaller influencer pool but may not yield a better solution.



(a) Ego-Facebook (b) Git

Fig. 6. Influence maximization with different seed set size.

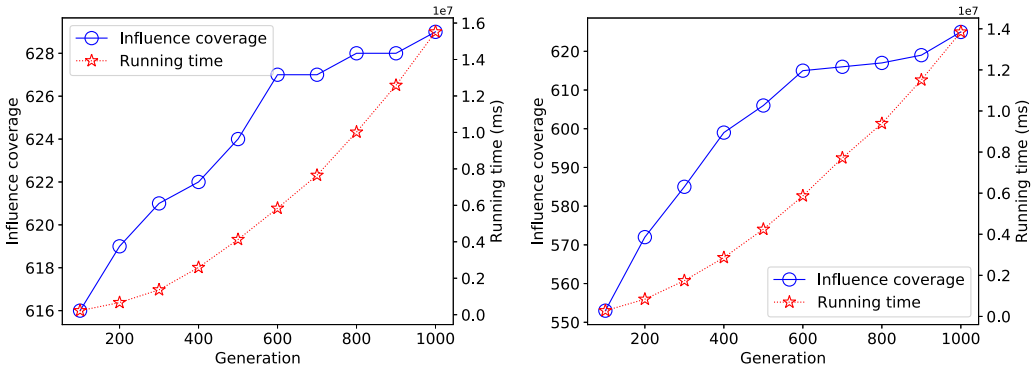


Fig. 7. ABEM performance with different generation numbers.

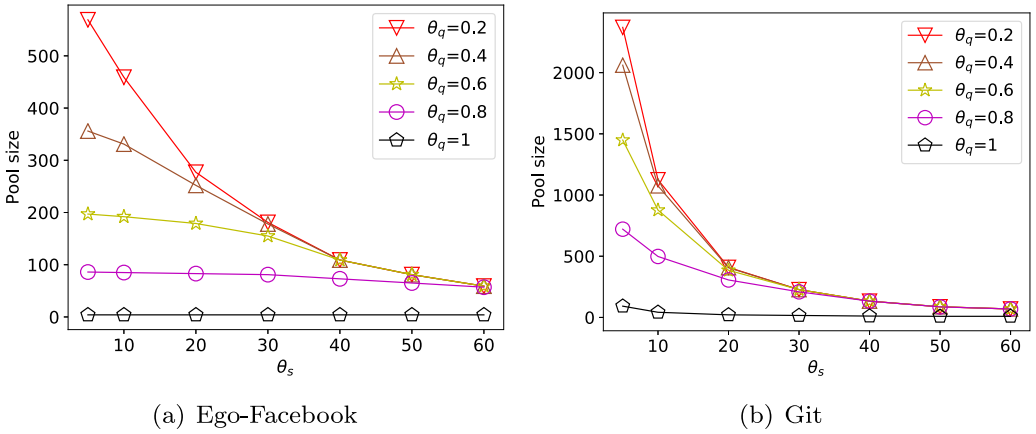


Fig. 8. Pool size with different parameter settings.

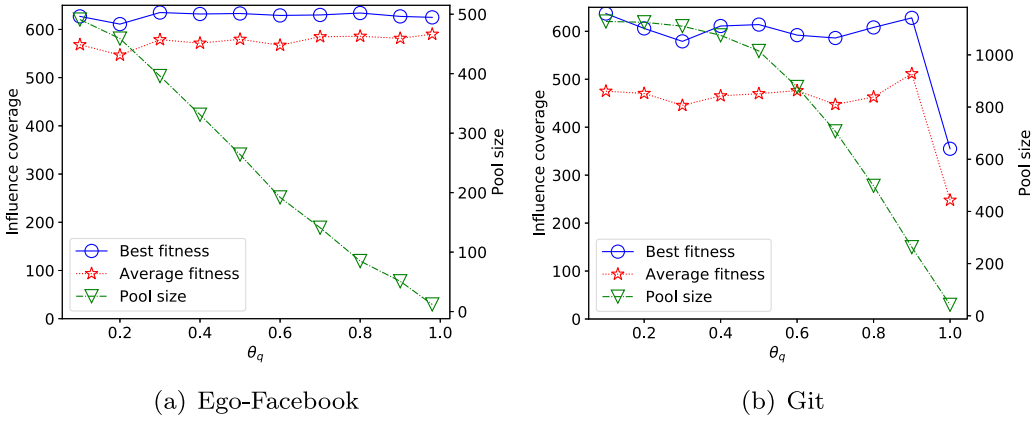


Fig. 9. Pool size analysis.

This is due to the reason that potential influencers may be filtered out when decreasing the size of the influencer pool.

Figs. 9(a) and 9(b) demonstrate the influencer pool size analysis by using two datasets. In the Ego-Facebook dataset, no significant performance improvement can be observed when decreasing the influencer pool size. In other words, the smallest size enables the ABEM to yield almost the same performance as that of a larger pool size. This phenomenon also implies ABEM will carry out a similar seed set as that of the degree-based selection in the Ego-Facebook network. This is also consistent with the results of Experiment 2. By contrast, Fig. 9(b) reveals a different pattern. The best fitness shows a steady trend until θ_q reaches 0.9. Starting from this point, both best and average fitness drop dramatically. This is because potential influencers are filtered out due to narrowing down the search scope. Therefore, by considering both efficiency and effectiveness, ABEM will adopt $\theta_q = 0.9$ with an

influencer pool size of 600. In this case, ABEM will definitely outperform degree-based selection, which is also consistent with the results of Experiment 2.

6. Conclusion

In this paper, a novel agent-based evolutionary approach, i.e., ABEM, is proposed to mine influencers in online social networks. We elaborate on the proposed approach, including algorithms, mining process and each component of ABEM in detail. We also clarify the major capabilities of ABEM, i.e., handling large-scale networks and dynamic environments. The former relies on agent-based modeling, where the major computational cost can be distributed to the individual agent in ABEM. The behavioral outcome provides a reasonable search scope for ABEM. Also, agent-based modeling enables ABEM to identify potential influencers in a distributed manner, which is suitable for real-world situations where the network changes without capturing any snapshots. The search scope of ABEM is updated by the user agents. The latter is handled by the proposed algorithms, which can retain the existing potential influencers and modify part of the solutions. Extensive experiments are conducted to evaluate the performance and capability of ABEM, including convergence analysis, continuous influence maximization in dynamic networks, classical influence maximization comparison and parameter analysis. The experimental results demonstrate that ABEM not only outperforms the state-of-the-art algorithms in the performance of influence maximization but also can be applied to a large-scale and dynamic environment.

In the future, we plan to further improve ABEM by employing some heuristics, which can expedite the convergence speed in a changing environment. Furthermore, we will develop an enhanced version of ABEM to fit a more sophisticated influence diffusion process with the consideration of context.

CRediT authorship contribution statement

Weihua Li: Conceptualization, Methodology, Software, Validation, Supervision, Writing – review & editing. **Yuxuan Hu:** Investigation, Visualization, Data curation, Writing – original draft. **Chenting Jiang:** Project administration, Writing – review & editing. **Shiqing Wu:** Formal analysis, Visualization, Writing – review & editing. **Quan Bai:** Conceptualization, Supervision, Writing – review & editing. **Edmund Lai:** Methodology, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All our research data comes from public datasets.

References

[1] D. Kempe, J. Kleinberg, É. Tardos, Maximizing the spread of influence through a social network, in: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 137–146.

[2] A. Likhyani, S. Bedathur, P. Deepak, Location-specific influence quantification in location-based social networks, ACM Trans. Intell. Syst. Technol. 10 (3) (2019) 1–28.

[3] N. Sumith, B. Annappa, S. Bhattacharya, Influence maximization in large social networks: Heuristics, models and parameters, Future Gener. Comput. Syst. 89 (2018) 777–790.

[4] J.S. Zhang, Q. Lv, Understanding event organization at scale in event-based social networks, ACM Trans. Intell. Syst. Technol. 10 (2) (2019) 1–23.

[5] J. Zhou, J. Fan, J. Wang, X. Wang, L. Li, Cost-efficient viral marketing in online social networks, World Wide Web 22 (6) (2019) 2355–2378.

[6] J. Zhou, Y. Zhang, J. Cheng, Preference-based mining of top-K influential nodes in social networks, Future Gener. Comput. Syst. 31 (2014) 40–47.

[7] S. Shelke, V. Attar, Source detection of rumor in social network–a review, Online Soc. Netw. Media 9 (2019) 30–42.

[8] L. Zhu, Y. Wang, Rumor diffusion model with spatio-temporal diffusion and uncertainty of behavior decision in complex social networks, Phys. A 502 (2018) 29–39.

[9] W. Tang, L. Tian, X. Zheng, G. Luo, Z. He, Susceptible user search for defending opinion manipulation, Future Gener. Comput. Syst. 115 (2021) 531–541.

[10] X. Liu, D. He, C. Liu, Information diffusion nonlinear dynamics modeling and evolution analysis in online social network based on emergency events, IEEE Trans. Comput. Soc. Syst. 6 (1) (2019) 8–19.

[11] M. Azaouzi, W. Mnasri, L.B. Romdhane, New trends in influence maximization models, Comp. Sci. Rev. 40 (2021) 100393.

[12] W. Li, Q. Bai, L. Liang, Y. Yang, Y. Hu, M. Zhang, Social influence minimization based on context-aware multiple influences diffusion model, Knowl.-Based Syst. 227 (2021) 107233.

[13] M. Huiyu, C. Jiuxin, Y. Tangfei, B. Liu, Topic based time-sensitive influence maximization in online social networks, World Wide Web 23 (3) (2020) 1831–1859.

[14] K. Konotopska, G. Iacca, Graph-aware evolutionary algorithms for influence maximization, 2021, arXiv preprint arXiv:2104.14909.

[15] G.G. Piva, F.L. Ribeiro, A.S. Mata, Networks with growth and preferential attachment: modelling and applications, J. Complex Netw. 9 (1) (2021) cnab008.

[16] M. Alam, M. Khan, K.S. Perumalla, M. Marathe, Generating massive scale-free networks: Novel parallel algorithms using the preferential attachment model, ACM Trans. Parallel Comput. 7 (2) (2020) 1–35.

[17] M.L. Bertotti, G. Modanese, The configuration model for barabasi-albert networks, Appl. Netw. Sci. 4 (1) (2019) 1–13.

[18] J. Ding, W. Sun, J. Wu, Y. Guo, Influence maximization based on the realistic independent cascade model, Knowl.-Based Syst. 191 (2020) 105265.

[19] H. Zhuang, Y. Sun, J. Tang, J. Zhang, X. Sun, Influence maximization in dynamic social networks, in: 2013 IEEE 13th International Conference on Data Mining, 2013, pp. 1313–1318.

[20] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '07, 2007, pp. 420–429.

[21] A. Goyal, W. Lu, L.V. Lakshmanan, Celf++ optimizing the greedy algorithm for influence maximization in social networks, in: Proceedings of the 20th International Conference Companion on World Wide Web, 2011, pp. 47–48.

[22] X. Tang, X. Liu, Improved evolution algorithm that guides the direction of individual mutation for influence maximization in social networks, in: International Conference on Knowledge Science, Engineering and Management, Springer, 2021, pp. 532–545.

[23] K. Zhang, H. Du, M.W. Feldman, Maximizing influence in a social network: Improved results using a genetic algorithm, Phys. A 478 (2017) 20–30.

[24] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 199–208.

[25] S. Chen, K. He, Influence maximization on signed social networks with integrated pagerank, in: 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), IEEE, 2015, pp. 289–292.

[26] X. Yin, X. Hu, Y. Chen, X. Yuan, B. Li, Signed-PageRank: An efficient influence maximization framework for signed social networks, IEEE Trans. Knowl. Data Eng. (2019).

[27] M. Gong, J. Yan, B. Shen, L. Ma, Q. Cai, Influence maximization in social networks based on discrete particle swarm optimization, Inform. Sci. 367 (2016) 600–614.

[28] S.S. Singh, A. Kumar, K. Singh, B. Biswas, LAPSO-IM: A learning-based influence maximization approach for social networks, Appl. Soft Comput. 82 (2019) 105554.

[29] S.S. Singh, K. Singh, A. Kumar, B. Biswas, ACO-IM: maximizing influence in social networks using ant colony optimization, Soft Comput. 24 (13) (2020) 10181–10203.

[30] H. Li, R. Zhang, X. Liu, An efficient discrete differential evolution algorithm based on community structure for influence maximization, Appl. Intell. (2022) 1–19.

[31] Y.-C. Chen, W.-Y. Zhu, W.-C. Peng, W.-C. Lee, S.-Y. Lee, CIM: Community-based influence maximization in social networks, ACM Trans. Intell. Syst. Technol. 5 (2) (2014) http://dx.doi.org/10.1145/2532549.

[32] M. Jaouadi, L.B. Romdhane, Din: an efficient algorithm for detecting influential nodes in social graphs using network structure and attributes, in: 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications, AICCSA, IEEE, 2016, pp. 1–8.
[33] M. Jaouadi, L.B. Romdhane, Influence maximization problem in social networks: An overview, in: 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications, AICCSA, IEEE, 2019, pp. 1–8.
[34] P. Krömer, J. Nowaková, Guided genetic algorithm for information diffusion problems, in: 2018 IEEE Congress on Evolutionary Computation, CEC, 2018, pp. 1–8.
[35] E. Bonabeau, Agent-based modeling: Methods and techniques for simulating human systems, Proc. Natl. Acad. Sci. 99 (suppl 3) (2002) 7280–7287.
[36] C.M. Macal, M.J. North, Agent-based modeling and simulation, in: Proceedings of the 2009 Winter Simulation Conference, WSC, 2009, pp. 86–98.
[37] P.-P. van Maanen, B. van der Vecht, An agent-based approach to modeling online social influence, in: Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2013, pp. 600–607.
[38] W. Li, Q. Bai, M. Zhang, A multi-agent system for modelling preference-based complex influence diffusion in social networks, Comput. J. 62 (3) (2018) 430–447.
[39] W. Li, Q. Bai, M. Zhang, Agent-based influence propagation in social networks, in: 2016 IEEE International Conference on Agents, ICA, 2016, pp. 51–56.
[40] W. Li, Q. Bai, T.D. Nguyen, M. Zhang, Agent-based influence maintenance in social networks, in: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, 2017, pp. 1592–1594.
[41] D. Bucur, G. Iacca, Influence maximization in social networks with genetic algorithms, in: Applications of Evolutionary Computation, 2016, pp. 379–392.
[42] R. Gopal, B. Rosmaita, D. Van Gucht, Genetic algorithms for the traveling salesman problem, in: Proceedings of the First International Conference on Genetic Algorithms and their Applications, vol. 160, 1984, pp. 160–168.
[43] A. Panizo-LLedot, G. Bello-Orgaz, D. Camacho, A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme, Future Gener. Comput. Syst. 110 (2020) 960–975.
[44] S. Banerjee, M. Jenamani, D.K. Pratihar, A survey on influence maximization in a social network, Knowl. Inf. Syst. 62 (9) (2020) 3417–3455.
[45] S. Peng, Y. Zhou, L. Cao, S. Yu, J. Niu, W. Jia, Influence analysis in social networks: A survey, J. Netw. Comput. Appl. 106 (2018) 17–32.
[46] Y. Li, J. Fan, Y. Wang, K.-L. Tan, Influence maximization on social graphs: A survey, IEEE Trans. Knowl. Data Eng. 30 (10) (2018) 1852–1872.
[47] R. Bian, Y.S. Koh, G. Dobbie, A. Divoli, Identifying top-k nodes in social networks: a survey, ACM Comput. Surv. 52 (1) (2019) 1–33.
[48] M. Han, M. Yan, Z. Cai, Y. Li, X. Cai, J. Yu, Influence maximization by probing partial communities in dynamic online social networks, Trans. Emerg. Telecommun. Technol. 28 (4) (2017) e3054.
[49] G. Tong, W. Wu, S. Tang, D.-Z. Du, Adaptive influence maximization in dynamic social networks, IEEE/ACM Trans. Netw. 25 (1) (2016) 112–125.
[50] Y. Wang, Q. Fan, Y. Li, K.-L. Tan, Real-time influence maximization on dynamic social streams, 2017, arXiv preprint arXiv:1702.01586.
[51] T. Murata, H. Koga, Extended methods for influence maximization in dynamic networks, Comput. Soc. Netw. 5 (1) (2018) 1–21.
[52] W. Li, K. Zhong, J. Wang, D. Chen, A dynamic algorithm based on cohesive entropy for influence maximization in social networks, Expert Syst. Appl. 169 (2021) 114207.
[53] N. Hafiene, W. Karoui, L. Ben Romdhane, Influential nodes detection in dynamic social networks: A Survey, Expert Syst. Appl. 159 (2020) 113642.
[54] C. Tsai, Y. Yang, M. Chiang, A genetic NewGreedy algorithm for influence maximization in social network, in: 2015 IEEE International Conference on Systems, Man, and Cybernetics, 2015, pp. 2549–2554.
[55] P. Krömer, J. Nowaková, Guided genetic algorithm for the influence maximization problem, in: International Computing and Combinatorics Conference, Springer, 2017, pp. 630–641.
[56] S. Agarwal, S. Mehta, Social influence maximization using genetic algorithm with dynamic probabilities, in: 2018 Eleventh International Conference on Contemporary Computing (IC3), IEEE, 2018, pp. 1–6.
[57] L. Cui, H. Hu, S. Yu, Q. Yan, Z. Ming, Z. Wen, N. Lu, DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks, J. Netw. Comput. Appl. 103 (2018) 119–130.
[58] S. Wang, X. Tan, Determining seeds with robust influential ability from multi-layer networks: A multi-factorial evolutionary approach, Knowl.-Based Syst. 246 (2022) 108697.
[59] J.J. Lotf, M.A. Azgomi, M.R.E. Dishabi, An improved influence maximization method for social networks based on genetic algorithm, Phys. A 586 (2022) 126480.
[60] W. Li, Q. Bai, M. Zhang, Siminer: A stigmergy-based model for mining influential nodes in dynamic social networks, IEEE Trans. Big Data 5 (2) (2018) 223–237.
[61] P. Panzarasa, T. Opsahl, K.M. Carley, Patterns and dynamics of users' behavior and interaction: Network analysis of an online community, J. Am. Soc. Inf. Sci. Technol. 60 (5) (2009) 911–932.
[62] B. Rozemberczki, C. Allen, R. Sarkar, Multi-scale attributed node embedding, 2019, arXiv:1909.13021.
[63] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, 2010, pp. 135–142.
[64] W. Li, Q. Bai, C. Jiang, M. Zhang, Stigmergy-based influence maximization in social networks, in: Proceedings of Pacific Rim International Conference on Artificial Intelligence, 2016, pp. 750–762.