Received June 25, 2018, accepted July 28, 2018, date of publication August 7, 2018, date of current version August 28, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2864240

LAIM: A Linear Time Iterative Approach for Efficient Influence Maximization in Large-Scale Networks

HONGCHUN WU[1,2], JIAXING SHANG[1,2], SHANGBO ZHOU[1,2], YONG FENG[1,2], BAOHUA QIANG[3,4], AND WU XIE[5]

[1]College of Computer Science, Chongqing University, Chongqing 400044, China
[2]Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing University, Chongqing 400044, China
[3]Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guilin University of Electronic Technology, Guilin 541004, China
[4]Guangxi Colleges and Universities Key Laboratory of Cloud Computing and Complex Systems, Guilin University of Electronic Technology, Guilin 541004, China
[5]Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

Corresponding author: Jiaxing Shang (shangjx@cqu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1402400, in part by the National Natural Science Foundation of China under Grant 61702059, in part by the China Postdoctoral Science Foundation under Grant 2017M612913, in part by the Chongqing Industrial Generic Technology Innovation Program under Grant cstc2017zdcy-zdzxX0010, in part by the Chongqing Postdoctoral Foundation under Grant Xm2017125, in part by the Frontier and Application Foundation Research Program of Chongqing City under Grants cstc2018jcyjAX0340 and cstc2017jcyjAX0340, in part by the Guangxi Key Laboratory of Trusted Software under Grant kx201702, and in part by the Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guangxi Colleges and Universities Key Laboratory of Cloud Computing and Complex Systems, under Grant YD17X04.

ABSTRACT The problem of influence maximization (IM) has been extensively studied in recent years and has many practical applications such as social advertising and viral marketing. Given the network and diffusion model, IM aims to find an influential set of seed nodes so that targeting them as diffusion sources will trigger the maximum cascade of influenced individuals. The largest challenge of the IM problem is its NP-hardness, and most of the existing approaches are with polynomial time complexity, making themselves unscalable to very large networks. To address this issue, in this paper, we propose LAIM: a linear time iterative approach for efficient IM on large-scale networks. Our framework has two steps: 1) influence approximation and 2) seed set selection. In the first step, we propose an iterative algorithm to compute the local influence of a node based on a recursive formula and use the local influence to approximate its global influence. In the second step, the k influential seed nodes are mined based on the approximated influence in the first step. Based on our model, we theoretically prove that the proposed approach has linear time and space complexity. We further accelerate our algorithm with simple modifications and propose its fast version. Experimental results on eight real-world large-scale networks exhibit the superiority of our approach over the state-of-the-art methods in terms of both effectiveness and efficiency.

INDEX TERMS Influence maximization, iterative algorithm, social networks analysis, information diffusion, computational complexity.

I. INTRODUCTION

Influence maximization (IM) is an extensively studied optimization problem in social networks analysis and big data mining. The research of this problem has great practical values in a lot of scenarios. For example, in social marketing, a company may choose the key opinion leaders to help advertising their new products or services. Political leaders become more and more active on social media platform such as Facebook and Twitter, so that their policy can be widely popularized through the public. On social networks, social influence can play a significant role to affect people's decision making [1]. In summary, given the network and diffusion model, influence maximization aims to find the best set of k individuals from the network such that targeting them as diffusion sources will trigger the maximum cascade of influenced individuals with the ''world-of-mouth'' effect [2].

The study of influence maximization has wide applications including viral marketing [3], [4], popularity prediction [5], [6], decision making [1], etc.

A. DIFFUSION MODEL

The research of influence maximization cannot be separated from the underlying diffusion models [7], which play a vital role. Two widely considered models are the linear threshold (LT) model and the independent cascade (IC) model [8], both of which are time-discrete. In these diffusion models, a node is associated with two states, namely, *active* or *inactive*. Active means the node has been successfully influenced, e.g., adopted a new product or accepted a new idea. If a node is active, it will propagate the influence to its inactive neighbors. Inactive means the influence has not yet reached the node or the influence has reached but the node refuses to be influenced. At the beginning all nodes in the network are in the *inactive* state, then k seed nodes are activated as diffusion sources and influence starts propagating from them, until the diffusion process finally tops.

1) IC MODEL

Under the IC model, each directed edge (u, v) of the network is associated with a propagation probability p_{uv}. At step t, for each active vertex u, it will try to activate each of its inactive neighbor, suppose v, and succeed with probability p_{uv}. Once v is successfully activated, it will remain active during the rest of the time, and from the next step $t + 1$, it will pass the influence to its neighbors in the same manner. On the contrary, if u falls to activate v, u will not make any more attempts in the future, i.e., edge (u, v) can be regarded as blocked. If v has multiple active neighbors, their attempts will be independent of each other. The propagation stops when no more individuals can be further activated.

2) LT MODEL

Under the LT model, whether an individual can be activated depends on its active neighbors in the current step. For an inactive vertex v, each of its adjacent incoming edge, suppose (u, v), has a weight parameter b_{uv} indicating how v is influenced by u. The weight parameters should satisfy the constraint $\sum_{u \text{ neighbor of } v} b_{uv} \leq 1$ for each v. Under this setting, each node v is given an activation threshold θ_v, which is usually randomly selected from the interval [0,1] at the beginning of diffusion. At step t, for any inactive node v, once the condition $\sum_{u \text{ active neighbor of } v} b_{uv} \geq \theta_v$ is achieved, v will be activated and remain active from step $t + 1$. The propagation stops until no more individuals can be further activated.

B. INFLUENCE MAXIMIZATION

Kempe *et al.* [8] first formulated the influence maximization (**IM**) problem as a discrete optimization problem, i.e., given a social graph $G(V, E)$ and the diffusion model, IM aims to find a subset $S \subseteq V$ of k seed nodes such that targeting them as diffusion sources will trigger the maximum number of influenced individuals under the diffusion model, i.e., $S^* = \arg_{S \subseteq V \wedge |S|=k} \max \sigma(S)$, where $\sigma(S)$ is an objective function (usually called *influence spread*) evaluating the number of influenced individuals. Since most diffusion models are stochastic, Kempe *et al.* [8] defined $\sigma(S)$ as the expected number of active individuals after the influence propagation stops. Based on the above definition, they proved that under both IC and LT models, the IM problem is NP-hard, and its objective function is submodular. Here a function $\sigma(\cdot)$ which maps subsets $S \subseteq V$ to non-negative real numbers, i.e., $\sigma : 2^V \rightarrow R$, is called submodular if it has the following two properties:

(1) $\sigma(S)$ is monotone increasing with S, i.e., $\forall S \subseteq T, \sigma(S) \leq \sigma(T)$;

(2) $\sigma(S)$ has the "diminishing returns" property, i.e., $\forall v \in V,\ S \subseteq T$, it satisfies that $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(T \cup \{v\}) - \sigma(T)$.

The "diminishing returns" property reflects the phenomenon that when we add an element to a set S, its marginal gain with respect to S, i.e., $\sigma(S \cup \{v\}) - \sigma(S)$ is at least as much as the marginal gain of adding the same element to a super set of S.

The IM problem has two major challenges. The first challenge, as given by Kempe *et al.* [8], is its NP-hardness. The second challenge, as shown by Chen *et al.* [9], is the #P-hardness in computing the exact value of the objective function $\sigma(S)$ for any given seed set S. Luckily, benefit from the theory of submodular functions [10], the first challenge can be addressed through a "hill-climbing" greedy algorithm. For the second challenge, a conventional way to solve it is by running Monte-Carlo (MC) simulations to approximate the objective function $\sigma(S)$. It was proved by Kempe *et al.* [8] that if one uses MC simulations to approximate the objective function, then the "hill-climbing" greedy algorithm can generate solutions with a factor of $(1-1/e-\varepsilon)$ approximation ratio ($\approx 63\%$) to the optimal solution, where e is the natural logarithm base and ε is an arbitrarily small positive number, which can be controlled by the number of MC simulations. Unfortunately, running MC simulations to approximate the objective function is a time consuming thing, as Chen *et al.* [11] had previously shown, the time complexity of the greedy algorithm with MC simulations can be as high as $O(knRm)$, where n and m are the number of nodes and edges, R is the number of MC simulations. Previous experimental studies showed that one usually needs tens of thousands of MC simulations to get a satisfactory approximation to the objective function. Consequently, the conventional greedy algorithm can only handle small networks.

In recent years, scholars proposed numerous methods to address the challenges of the IM problem. These approaches can be divided into several groups, e.g., simulation-based algorithms [12], [13], node centrality-based algorithms [14], [15], influence path-based methods [16], [17], reverse influence sampling-based algorithms [18], [19], etc. However, these approaches either cannot handle large networks with linear time/space complexity, or produce poor solutions with low accuracy. To the best of our knowledge, there are almost no influence maximization algorithms which not only provide high quality solutions, but also has linear time and space complexity.

Motivated by the above observations, in this article, we propose LAIM: a **L**inear time iterative **A**pproach for efficient **I**nfluence **M**aximization on large-scale social networks. Our framework has two steps: (1) influence approximation; and (2) seed set selection. In the first step we propose an iterative algorithm to compute the local influence of a node based on a recursive formula, and use the local influence to approximate its global influence. In the second step, the k influential seed nodes are mined based on the approximated influence in the first step. Based on our model, we theoretically prove the proposed approach has linear time and space complexity. We further accelerate our algorithm with simple modifications and propose its fast version. We conduct extensive experimental study on eight real-world large-scale networks and the results exhibit the superiority of our approach over the state-of-the-art methods, as evaluated by influence spread, memory usage and running time.

A preliminary conference version of our work appears in [20]. Compared to the conference version, this article has the following new contributions:

- A comprehensive literature review is provided, which covers most of the related research works published in the latest two years.
- The proposed approach is more detailedly illustrated and analyzed, both theoretically and experimentally.
- The evaluation framework is largely enriched by incorporating more datasets, including more competitive baselines, and conducting more experiments, together with performance analysis, which provides a deep insight into our proposed approach.

The rest of this article has the following parts. Section II presents a literature review on the study of influence maximization. Section III introduces the preliminaries of the problem and our model. In Section IV we propose the linear time iterative approach. Section V presents the evaluation framework, including the datasets, baseline methods, evaluation measures, and experimental environment. In Section VI we present and discuss the results. Section VII concludes this research.

II. LITERATURE REVIEW

Numerous research works have been proposed in recent years to solve the influence maximization problem. We roughly divide these studies into four groups: (1) simulation-based methods, (2) node centrality-based methods, (3) influence path-based methods, and (4) reverse influence sampling (RIS) based methods.

A. SIMULATION-BASED METHODS

The general idea behind simulation-based methods is to design new tricks to reduce the number of MC simulations without compromise on solution accuracy.

Leskovec *et al.* [12] proposed CELF algorithm to improve the time efficiency of the traditional greedy algorithm through a lazy forward strategy. In their approach, they maintained the marginal gains of individuals with respect to the current seed set and partially updated their marginal gains (through MC simulations) only when it was necessary. Through experiments the authors showed their approach was much more efficient than the traditional greedy algorithm and it can generate solutions with the same approximation ratio to the optimal solution.

Inspired by similar idea, Goyal *et al.* [13] proposed CELF++, an improved version of CELF. Compared to CELF, CELF++ not only stored the marginal gains of individuals corresponding to the current seed set, but also maintained their marginal gains with respect to a super set of the seed set, which further reduced the unnecessary checking of candidates. In their experiments the CELF++ algorithm achieved about 30% $\sim$ 50% speed up over the CELF algorithm.

Zhou *et al.* [21] proposed a new upper bound of the influence spread under IC and LT models. The new upper bound was then utilized to prone the MC simulations of the greedy algorithm. Experimental results showed their approach achieved about 2 $\sim$ 10 times speedup as compared with the CELF algorithm.

A common disadvantage shared by the simulation-based algorithms is that they all require a large number of MC simulations to obtain a close approximation to the objective function, making their worst-case time complexity as much as $O(knmR)$, thus cannot scale to large networks.

B. NODE CENTRALITY-BASED METHODS

Node centrality-based algorithms mine the influential seed nodes according to some centrality measures defined on a node or a set of nodes.

Chen *et al.* [11] proposed the SD (Single Discount) algorithm by taking into consideration the impact of previously chosen seeds on current candidates. Specifically, the algorithm iteratively selected a new seed node with the maximum degree. When a node was chosen, the degree of each of its neighbor will be decreased (discounted) by a unit, in order to avoid influence overlap. Experimental results showed that the SD algorithm generated more accurate solutions than naively selecting the top seeds with the highest degrees.

Liu *et al.* [14] proposed a Group-PageRank based algorithm, where the influence from seed set S to a node v is modeled as the combination of influence of its neighbors, which can be recursively computed. Based on the IC model, the authors proposed a compact upper bound of the influence spread function and used the lazy-forward heuristic to find the best seeds.

Recently, Zhu *et al.* [22] proposed SHIM, a structure hole-based influence maximization algorithm which utilized structure hole as the centrality measurement. They first identified structure hole spanners whose structure hole value was above the given threshold, followed by computing the influence capability of each structure hole spanner. After that, they selected the top-k seeds by combining the structure hole value and influence value.

Liu *et al.* [15] proposed a closeness centrality-based algorithm, where they defined a generalized closeness centrality (GCC) index by generalizing the closeness centrality index from a single node to a set of nodes.

Riquelme *et al.* [23] mainly considered the centrality measures under the LT diffusion model, and proposed a new centrality measure named LTR (Linear Threshold Rank) to rank the users of the network.

Some centrality based algorithms have linear time complexity, for example, the time complexity of the naive degree-based algorithm is $O(m + n \log n)$. However, since they take little consideration of the diffusion process, they usually generate inaccurate solutions. Other centrality-based methods may have high time complexity. For example, the the structure-hole based algorithm [22] requires $O(n^3)$ time to compute the structure-hole values, the closeness-centrality based method [15] needs to determine the Laplacian matrix and calculate its eigenvectors, making themselves unable to scale to large networks.

C. INFLUENCE PATH-BASED METHODS

The general idea behind influence path-based algorithms is to reduce the randomness of diffusion models by restricting influence propagation on specific paths, such that the objective function can be efficiently calculated without MC simulations.

Chen *et al.* [9], [24] studied influence path-based methods under both IC and LT models. For the IC model, they proposed MIA (Maximum Influence Arborescence) and PMIA (Prefix excluding MIA) [9], both of which were based on arborescence, a local tree structure defined for a node to approximate its influence area. Specifically, it was a node-centralized subgraph where the influence propagation from the center node to other individuals in the subgraph was restricted on the paths with the maximum propagation probabilities. The algorithm iteratively selected new seed nodes and updated the arborescence structure for other candidates, until k seeds were found. For the LT model, they proposed LDAG (Local Directed Acyclic Graphs) [24], which was a local graph structure centered around a node to represent its local influence area. Based on LDAG, the influence of a candidate can be recursively computed without MC simulations. The seed selection procedure was similar to MIA and PMIA.

One limitation of Chen *et al.*'s approach [9] is that they only considered the influence paths with the maximum propagation probabilities, which may affect the solution accuracy on some datasets. To address this issue, Kim *et al.* [16] proposed IPA (Independent Path Algorithm), an influence path-based method which simultaneously considered multiple influence paths and assumed that their propagations were independent of each other. Consequently, the overall influence spread can be more efficiently calculated in a distributed way and the objective function can be more accurately approximated.

Liu *et al.* [17] proposed influence path-based algorithms to solve the time-constrained influence maximization problem. In their approach they first constructed influence spreading paths, which were defined as the paths from seed nodes to non-seed nodes. The length of a path was the cumulated time delay of edges on that path, while the probability of a path was the product of propagation probabilities of edges constituting the path. Then the paths with length longer than T or probability smaller than θ were pruned to reduce the computational overload. Based on the influence spreading paths, they further proposed a parallel algorithm to mine the seed nodes.

Ko *et al.* [25], [26] considered the IC model and proposed a target-oriented estimation (TOE) method for influence maximization, which can remedy the drawback of existing path-based algorithms. Unlike previous path-based methods which utilize influence path to approximate the amount of influence a seed node can exert to non-seed nodes, their approach focused on the amount of influence a non-seed node can receive from a given seed set. They showed their approach achieved higher accuracy as compared to existing path-based algorithms.

Compared with simulation and centrality-based algorithms, influence path-based algorithms reach a tradeoff between running time and solution quality. However, a critical deficiency of these methods is that they usually require a huge amount of memory to maintain the influence paths, which restricts their space efficiency, making themselves unscalable to very large networks.

D. RIS-BASED ALGORITHMS

Under the IC model, Borgs *et al.* [27] proposed the first RIS (Reverse Influence Sampling) based algorithm, which used "node-centric" sampling strategies to approximate the objective function. The general framework is: select a node v uniformly at random, and determine the set of nodes that would have influenced v. If this process is repeated multiple times, and a certain node u appears often as an "influencer", then u is likely a good candidate for the most influential node. The authors theoretically proved that the probability a node u appeared in a set of influencers was proportional to its expected influence on the network. To compute the influence objective function, one only had to repeat the sampling process multiple times until an approximation guarantee was achieved.

Tang *et al.* [18] proposed the TIM+ (Two-phase Influence Maximization) algorithm which generalized the RIS framework from IC model to both IC and LT models. Their approach consisted of two phases where the first phase estimated an approximation parameter θ while the second phase generated θ samples to derive to best seed set. The algorithm has a parameter ϵ which controls its accuracy. Tang *et al.* [19] further proposed the improved version of TIM+, i.e., IMM (Influence Maximization via Martingales) algorithm, which took advantage of martingales to improve its time and memory efficiency. Similarly, the accuracy of IMM was controlled by a parameter ϵ.

Based on the idea of RIS, Nguyen *et al.* [28] proposed the SSA (Stop and Stare Algorithm) and D-SSA (Dynamic SSA) algorithms which focused on characterizing the minimum number of RIS samples needed to achieve $(1 - e - \epsilon)$ approximation. The algorithm kept generating samples and *stopped* at exponential check points to verify (*stare*) if there was adequate statistical evidence on the solution quality for termination. The authors claimed that their approach largely outperforms the IMM algorithm. However, in a recent research, Huang *et al.* [29] conducted a rigorous theoretical and experimental analysis of SSA and D-SSA, and proposed a revised version.

Recently, Wang *et al.* [30] proposed a novel bottom-k sketch based RIS framework, namely BKRIS, which brought the order of samples into the RIS framework. By applying the sketch technique, they derived early termination conditions to significantly accelerate the seed set selection procedure.

Although the RIS based algorithms exhibit competitive performance in terms of effectiveness and efficiency, they still have their inherent drawbacks. Despite that these algorithms can find a seed set S with approximation guarantee, they cannot make sure that the subsets of S will also provide the same approximation guarantee.

E. OTHER STUDIES

Besides the above introduced methods, there are also many other influence maximization methods, such as the multiple selectors combination based algorithm [31], community-based algorithm [32], multi-objective optimization based method [33], competitive influence maximization [34], location-based influence maximization [35], [36], etc.

III. PRELIMINARIES

A. INFLUENCE MAXIMIZATION PROBLEM

Kempe *et al.* [8] first formulated influence maximization as the following discrete optimization problem:

Definition 1 (Influence Maximization, ***IM*** *[8]):* Given a social graph $G(V, E)$ where V is the set of nodes and E is the set of edges, an integer budget k, under a predefined diffusion model, the influence maximization problem aims to find a subset $S \subseteq V$ of k seed nodes such that targeting them as diffusion sources will trigger the maximum number of influenced individuals under the diffusion model, i.e.,

$$S^* = \arg_{S \subseteq V \wedge |S|=k} \max \sigma(S) \tag{1}$$

where $\sigma(S)$ is the *influence spread* defined as the overall number of activated individuals at the end of influence propagation triggered by S.

As we have introduced in Section I-B, Kempe *et al.* [8] proved that under both LT and IC models, the IM problem is NP-hard and its objective function, i.e., $\sigma(S)$, is submodular about S. So they further proposed a "hill-climbing" greedy algorithm to solve the problem, the pseudo code of which is shown in Algorithm 1. From the code we see that the

Algorithm 1 The Hill-Climbing Greedy Algorithm (Kempe *et al.* [8])

```
Require:
    Network $G$, number of seed nodes $k$
Ensure:
    Seed set $S$
1: Initialize: Let $S \leftarrow \Phi$.
2: for $i = 1$ to $k$ do
3:     $v = \arg_u \max\{\sigma(S \cup \{u\}) - \sigma(S)\}$
4:     // $\sigma(*)$ is computed using Monte-Carlo simulations
5:     $S \leftarrow S \cup \{v\}$
6: end for
7: return S
```

algorithm starts with $S = \Phi$, and then iteratively adds a new seed node which has the maximum marginal gain with respect to the current seed set, until k seed nodes are successfully discovered.

Kempe *et al.* [8] proved that the greedy algorithm provides a factor of $(1 - 1/e - \varepsilon)$ approximation ratio to the optimal solution, as shown in Theorem 1.

Theorem 1 (Kempe et al. [8]): For the influence maximization problem defined in Definition 1, the solution of Algorithm 1 provides a factor of $(1-1/e-\varepsilon)$ *approximation ratio to the optimal solution* S^*, *i.e.,* $\sigma(S) \geq (1 - 1/e - \varepsilon)\sigma(S^*)$.

As shown in Algorithm 1, the greedy algorithm requires MC simulations to approximate the objective function. Chen *et al.* [11] proved that the time complexity of the greedy algorithm is $O(knRm)$, as shown in Theorem 2.

Theorem 2 (Chen et al. [11]): For the influence maximization problem defined in Definition 1, if Monte-Carlo simulations are used to approximate the objective function, then the greedy algorithm defined in Algorithm 1 has time complexity $O(knRm)$, *where n and m are the number of nodes and edges, k is the number of seed nodes to be mined, and R is the number of MC simulations used to estimate* $\sigma(S)$.

B. NETWORK PRELIMINARIES

Without loss of generality, we define a directed social graph[1] as $G(V, E)$, where V is the set of nodes and E is the set of edges. Generally, G can be described through an adjacency matrix A, where:

$$A_{i,j} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

For any node v, its outgoing neighbor set is:

$$N(u) = \{v | A_{u,v} > 0\} \tag{3}$$

Before introducing our LAIM approach, we give the following definitions:

Definition 2 (γ-th Layer Neighbor): Given a network, for any node u and nonnegative integer $\gamma \geq 0$, we define u's γ-th

[1]Our approach is also applicable to undirected networks.

layer neighbor set as:

$$N_\gamma(u)=\begin{cases}\bigcup_{v\in N_{\gamma-1}(u)} N(v) & \gamma>0\\ \{u\} & \gamma=0\end{cases}\tag{4}$$

From definition 2 we can easily see that $N_0(u)=\{u\}$ and $N_1(u)=N(u)$. Intuitively, the γ-th layer neighbor set indicates the set of individuals whose distance to u is γ.

Definition 3 (γ Neighborhood): Given a network, for any node u and nonnegative integer $\gamma\geq 0$, node u's γ neighborhood $N_{\leq\gamma}(u)$ is defined as:

$$N_{\leq\gamma}(u)=\bigcup_{\ell=0}^{\gamma} N_\ell(u)\tag{5}$$

Intuitively, u's γ neighborhood represents the set of nodes whose distance to u is less than or equal to γ.

Definition 4 (Node Influence): For any node $u\in V$, its influence on network $G(V,E)$ is denoted by $I_G(u)$. Similarly, u's influence on network $G-v$ is denoted by $I_{G-v}(u)$, where $G-v$ is the subgraph induced from G by V's subset $V\setminus\{v\}$.

Definition 5 (Local Influence): For any node $u\in V$, its influence within its γ neighborhood $N_{\leq\gamma}(u)$ on network G and $G-v$ are denoted by $I_G^{\leq\gamma}(u)$ and $I_{G-v}^{\leq\gamma}(u)$, respectively. Similarly, u's influence to its γ-th layer neighbor $N_\gamma(u)$ on network G and $G-v$ are denoted by $I_G^\gamma(u)$ and $I_{G-v}^\gamma(u)$, respectively.

IV. LINEAR TIME ALGORITHM

In this article, we mainly focus on the IC model. Our LAIM framework has two steps: (1) influence approximation; and (2) seed set selection. In the first step we propose an iterative algorithm to compute the local influence of a node based on a recursive formula, and use the local influence to approximate its global influence. In the second step, the k influential seed nodes are mined based on the approximated influence in the first step. In the following part we will present our two step approach in detail.

A. INFLUENCE APPROXIMATION

The approximation of influence spread is motivated the following observation: for any node u, if it is selected as a seed node, then the influence will firstly propagate from u to its first layer neighbors $N(u)$, and then to its $2,3,\cdots,n$-th layer neighbors, i.e., $N_2(u),N_3(u),\cdots,N_n(u)$, until the influence propagation process finally stops. Under the IC diffusion model, in many real-world applications it is commonly observed that the propagation probabilities between individuals are small, so the influence cannot propagate far away from u, which means we may use the influence of u within its local neighborhood (local influence) to approximate its influence on the overall network.

Based on the above idea, given network $G(V,E)$, for any node $u\in V$ and nonnegative integer $\gamma\geq 0$, we approximate u's overall influence on G by:

$$I_G(u)\simeq I_G^{\leq\gamma}(u)=\sum_{\ell=0}^{\gamma} I_G^\ell(u)\tag{6}$$

Since u must first influence its first layer neighbor $N(u)$ in order to influence others nodes in $N_{\leq\gamma}(u)$, and from Eq. 4 we see that u's γ-th layer neighbors are included in the $\gamma-1$-th layer neighbors of $N(u)$, so we may represent $I_G^\gamma(u)$ (the influence of u to its γ-th layer neighbors) as a function of $N(u)$'s influence to their $\gamma-1$-th layer neighbors:

$$I_G^\gamma(u)=f\left(I_G^{\gamma-1}(v_1),I_G^{\gamma-1}(v_2),\cdots,I_G^{\gamma-1}(v_q)\right)\tag{7}$$

where $N(u)=\{v_1,v_2,\cdots,v_q\}$. In the IC diffusion model we know that node u will successfully activate each of its inactive neighbor v with probability p_{uv}, so we can further formulate $I_G^\gamma(u)$ as:

$$I_G^\gamma(u)=\sum_{v\in N(u)} p_{uv}\cdot I_G^{\gamma-1}(v)\tag{8}$$

Given that the $\gamma-1$-th layer neighbors of different nodes in $N(u)$ may overlap, the above formula is an approximate evaluation. In IC model, each node can be activated only once (no repeated activations allowed), so we should eliminate u from v's $\gamma-1$-th layer neighbor set in computing $I_G^{\gamma-1}(v)$ for any $v\in N(u)$ to avoid the re-computation of influence of u. As a result, $I_G^\gamma(u)$ can be further represented as:

$$I_G^\gamma(u)=\sum_{v\in N(u)} p_{uv}\cdot I_{G-u}^{\gamma-1}(v)\tag{9}$$

where $\gamma\geq 1$. New we prove that Eq. 9 can be effectively approximated by a recursive formula.

Theorem 3: Eq. 9 can be approximated by the following recursive formula:

$$I_G^\gamma(u)=\begin{cases}0 & \gamma=-1\\ 1 & \gamma=0\\ \sum\limits_{v\in N(u)} p_{uv}\left(I_G^{\gamma-1}(v)-p_{vu}I_G^{\gamma-2}(u)\right) & \gamma\geq 1\end{cases}\tag{10}$$

Proof: We give the proof by considering $\gamma=0$, $\gamma=1$ and $\gamma>1$ separately.

(1) If $\gamma=0$, since $N_0(u)=u$, node u only influence itself, so its influence is a unit.

(2) If $\gamma=1$, according to Eq. 10, it is easy to see that: $I_G^0(v)=I_{G-u}^0(v)=1$, so we have:

$$\begin{aligned}I_G^1(u)&=\sum_{v\in N(u)} p_{uv}\left(I_G^0(v)-p_{vu}I_G^{-1}(u)\right)\\&=\sum_{v\in N(u)} p_{uv}\cdot I_G^0(v)\\&=\sum_{v\in N(u)} p_{uv}\cdot I_{G-u}^0(v)\end{aligned}$$

Eq. 9 satisfies. Intuitively, when $\gamma=1$, we have $N_1(u)=N(u)$, so node u only influences its first layer neighbors, and activate each neighbor $v\in N(u)$ with probability p_{uv}.

(3) If $\gamma>1$, according to Eq. 10, we have:

$$I_G^\gamma(u)=\sum_{v\in N(u)} p_{uv}\left(I_G^{\gamma-1}(v)-p_{vu}I_G^{\gamma-2}(u)\right)$$

Algorithm 2 Influence Approximation

```
Require:
    Network $G(V, E)$, diffusion parameters $\{p_{uv}\}$, iterative parameter $\gamma$
Ensure:
    Local influence: $I_G^{\leq\gamma}(u), \forall u \in V$
1: Initialize: Let $I_G^{-1}(u) = 0$, $I_G^{0}(u) = 1$, $I_G^{\leq\gamma}(u) = 0$, $\forall u \in V$.
2: for $\ell = 1$ to $\gamma$ do
3:     for $u \in V$ do
4:         $I_G^{\ell}(u) = \sum_{v \in N(u)} p_{uv}\big(I_G^{\gamma-1}(v) - p_{vu} I_G^{\gamma-2}(u)\big)$
5:         $I_G^{\leq\gamma}(u) \leftarrow I_G^{\leq\gamma}(u) + I_G^{\ell}(u)$
6:     end for
7: end for
8: return $I_G^{\leq\gamma}(u), \forall u \in V$
```

By substituting $I_G^{\gamma-1}(v)$ according to Eq. 10 again, we further have:

$$\begin{aligned} I_G^{\gamma}(u) &= \sum_{v \in N(u)} p_{uv}\Big[\sum_{z \in N(v)\setminus\{u\}} p_{vz}\big(I_G^{\gamma-2}(z) - p_{zv} I_G^{\gamma-3}(v)\big) \\ &\quad + p_{vu}\big(I_G^{\gamma-2}(u) - p_{uv} I_G^{\gamma-3}(v)\big) - p_{vu} I_G^{\gamma-2}(u)\Big] \\ &\simeq \sum_{v \in N(v)} p_{uv}\Big[\sum_{z \in N(v)\setminus\{u\}} p_{vz}\big(I_G^{\gamma-2}(z) - p_{zv} I_G^{\gamma-3}(v)\big)\Big] \\ &= \sum_{v \in N(v)} p_{uv} \cdot I_{G-u}^{\gamma-1}(v) \end{aligned}$$

where the last equation of the formula is achieved by ignoring the smaller item $p_{vu}p_{uv} \cdot I_G^{\gamma-3}(v)$, so Eq. 9 satisfies.

In sum, Eq. 9 can be approximated by Eq. 10. □

Based on theorem 3, we give the following iterative algorithm to recursively compute the local influence of all nodes. Line 1 initializes the local influence. Line 2-7 iteratively calculate the local influence for all the nodes according to the recursive formula defined in Eq. 10.

B. SEED SET SELECTION

Based on the influence approximation algorithm of Algorithm 2, we further propose our **LAIM** algorithm which iteratively chooses top k influential seed nodes in a greedy manner. Specifically, the LAIM algorithm first applies Algorithm 2 to calculate the local influence for all individuals, base on which it further chooses the node with the maximum local influence $I_G^{\leq\gamma}(v)$ and add it to the seed set S. After that the selected seed node is removed from the social graph G and Algorithm 2 is re-executed to update the local influence of the rest individuals. After k seed nodes are successfully selected, the LAIM algorithm will stop and output the seed set. The pseudo code of LAIM is shown in Algorithm 3.

To further accelerate the algorithm, we provide the simple fast version of LAIM, namely, **FastLAIM**, which directly selects top k individuals with the maximum values of $I_G^{\leq\gamma}(v)$ calculated by Algorithm 2 in the first step and output them as the seed nodes.

Algorithm 3 The LAIM Algorithm

```
Require:
    Network $G(V, E)$, diffusion parameters $\{p_{uv}\}$, iterative parameter $\gamma$
Ensure:
    Seed set $S$
1: Initialize: Let $S \leftarrow \Phi$
2: for $i = 1$ to $k$ do
3:     Compute $I_G^{\leq\gamma}(u), \forall u \in V$ with Algorithm 2
4:     $v = \arg_{u \in V} \max\{I_G^{\leq\gamma}(u)\}$
5:     $S \leftarrow S \bigcup \{v\}$
6:     $V \leftarrow V \setminus \{v\}$
7: end for
8: return $S$
```

C. TIME AND SPACE COMPLEXITY

Here we give the time and space complexity analysis of our proposed approach.

Theorem 4 (Time Complexity): The time complexity of the LAIM algorithm is $O(k\gamma m)$, where $m = |E|$ is the number of edges, γ is the iterative parameter, and k is the number of seed nodes.

Proof: We first show the time complexity of local influence calculation. As show in Algorithm 2, the initialization in line 1 requires (n) time, where $n = |V|$ is the number of nodes. In each iteration, for any node $u \in V$, we need to traverse all the neighbors of u in order to compute $I_G^{\gamma}(u)$ according to formula 10. Traversing the neighbors of u requires $O(|N(u)|)$ time. So the time complexity of one round of iteration is $O(\sum_{u\in V} |N(u)|) = O(m)$. Since Algorithm 2 contains γ round of iterations, its time complexity is $O(\gamma m)$. From Algorithm 3 it is easy to see that the LAIM algorithm will repeat Algorithm 2 for k times to select the best k seed nodes. So the overall time complexity of the LAIM algorithm is $O(k\gamma m)$. □

Theorem 5 (Time Complexity): The time complexity of the FastLAIM algorithm is $O(\gamma m + n \log n)$, $n = |V|$ is the number of nodes, $m = |E|$ is the number of edges, and γ is the iterative parameter.

Proof: As we have shown in the former theorem, the time complexity of local influence approximation, i.e., Algorithm 2, is $O(\gamma m)$. After the local influence of all nodes are calculated, the FastLAIM algorithm first sorts the nodes according to their local influence, which requires $O(n \log n)$ time, and then directly output the top k nodes with the maximum local influence. So the overall time complexity of the FastLAIM algorithm is $O(\gamma m + n \log n)$. □

Theorem 6 (Space Complexity): If we use adjacency list to represent the network, then the space complexity of the LAIM algorithm will be $O(m + \gamma n)$, where $n = |V|$ is the number of

TABLE 1. Statistical properties and the parameter values (γ) of FastLAIM and LAIM of eight real world datasets.

Properties	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Pokec	LiveJournal	Orkut
# Ndoes	15K	37K	76K	335K	317K	1.6M	4M	3.1M
# Edges	31K	174K	406K	926K	1M	22.3M	35M	117M
Max. Degree	64	178	3,044	290	343	14,854	14,724	33,313
Avg. Degree	4.12	9.38	10.69	4.34	6.62	27.22	17.01	76.17
γ (FastLAIM)	4	5	5	4	7	6	6	5
γ (LAIM)	4	7	5	5	8	6	7	5

nodes, $m = |E|$ is the number of edges, and γ is the iterative parameter.

Proof: To store the network with adjacency list, we need $O(m)$ space. Since Algorithm 2 requires γ rounds of iterations, and each round of iteration needs $O(n)$ space to maintain the $I_G^{\gamma}(u)$ information of all nodes, the space complexity of Algorithm 2 is $O(\gamma n)$. So the overall space complexity of the LAIM algorithm is $O(m + \gamma n)$ which is necessarily $O(m)$ when $m \gg \gamma n$ □

From the above theorems it is easy to see that our proposed approach has linear time and space complexity.

V. EVALUATION

A. DATASET

We test our proposed approaches, i.e., LAIM and FastLAIM, on eight real-world datasets with different sizes and categories, including scholar cooperation networks, customer review networks, products co-selling networks, etc. We give a brief introduction about these datasets as follows:

NetHEPT: NetHEPT [11] is a scholar cooperation dataset from the arXiv electronic preprint platform.[2] The network consists of 15K nodes and 31K edges, where nodes represent the scholars in the High Energy Physics Theory area while edges represent the co-authorships among different scholars.

NetPHY: Similar to NetHEPT, NetPHY is another scholar cooperation dataset from the arXiv platform consisting of the co-authorships among researchers in the field of Physics. The dataset has 37K authors and 174K edges.

Epinions: Epinions [37] is a who-trust-whom social network from the customer reviewer site Epinions.com. If a customer trust the review of another one, an edge will be established between them. The network consists of 76K vertices and 406K edges, after all repeated edges were combined.

Amazon: The Amazon dataset was collected by Yang and Leskovec [38] from Amazon,[3] an online shopping site in America. If a product is frequently co-purchased by customers with another product, then an undirected edge is established between them. The network has 335K vertices and 926K links.

DBLP: DBLP is another dataset collected by Yang and Leskovec [38] from the DBLP platform[4] and provides co-authorships among scholars in the field of computer science. The network has 317K vertices and 1M edges where each edge indicates that the corresponding two scholars (vertices) have coauthored at least one paper.

Pokec: Pokec is a popular online social platform from Slovakia and the dataset covers the overall social relationships among Slovakia users. The network has 1.6M vertices and 22.3M edges, collected by Takac and Zabovsky [39].

LiveJournal: LiveJournal is a free online social platform where users can keep a blog, journal or diary. The users can also declare friendships with each other, forming the edges of the network. The dataset was collected by Yang and Leskovec [38] and has about 4M nodes and 35M edges.

Orkut: Orkut is another free online social platform which was previously operated by Google. It aims to help users maintain existing relations with old friends and establish new relations with new friends. This network, which consists of 3.1M users and 117M relations, was collected by Yang and Leskovec [38] and is the largest one in our experiments.

Most of our datasets are obtained form the Stanford large network dataset collection platform (SNAP).[5] For simplicity, we treat all the network graphs as undirected ones. The statistical properties of these datasets are summarized in Table 1, together with the parameter values (γ) of FastLAIM and LAIM. We experiment with different parameter values, and those with good performance in terms of both influence spread and running time are chosen and shown in Table 1.

B. BASELINE METHODS

We compare our approach with the following five baseline methods:

IMM: IMM [19] (Influence Maximization via Martingales) is an extension of the RIS-based TIM+ method proposed by Tang *et al.* [18]. It provides the same $(1 - 1/e - \varepsilon)$-approximation solution while significantly improves time efficiency by utilizing martingales. Compared to TIM+'s two-phase approach, IMM adds an extra intermediate step to heuristically refine θ into a tighter lower bound. The algorithm has a parameter ε controlling its accuracy. In this paper, we use the default setting $\epsilon = 0.1$ as the authors recommended in their paper.

BKRIS: BKRIS (Bottom-k sKetch-based RIS) is another RIS-based algorithm proposed by Wang *et al.* [30], which brought the order of samples into the RIS framework. By applying the sketch technique, they derived early

[2]http://www.arxiv.org/
[3]http://www.amazon.com/
[4]https://dblp.uni-trier.de/
[5]http://snap.stanford.edu/data/

termination conditions to significantly accelerate the seed set selection procedure. Compared to IMM, BKRIS has better time efficiency. Similar to IMM, we set $\epsilon = 0.1$ for the BKRIS algorithm.

CoFIM: The CoFIM (Community based Framework for Influence Maximization) is a community based influence maximization framework [32], which assumes that influence first propagates from seed nodes to their neighbors and then from these neighbors to other nodes within the same community. This algorithm has been proved to have good performance in large scale networks.

IPA: IPA [16] (Independent Influence Path Algorithm) is a path based influence maximization algorithm mainly designed for IC model. It approximates the influence function through independent influence paths whose propagation probability is higher than a threshold parameter θ. The propagation probability of an influence path is defined as the product of propagation probabilities along that path. As the authors did in their paper, we set $\theta = 1/320$.

TOE: TOE [26] (Target-Oriented Estimation) is another path based influence maximization algorithm. Unlike traditional methods' source-oriented strategy, TOE takes a target-oriented approach to estimate the influence spread received by each target node from the given seed set. As the authors declared, this approach can remedy the inaccuracy problem of existing path based methods. We set the threshold parameter $\theta = 1/160$, as recommended by the authors.

SD: The SD [11] (Single Discount) algorithm is a node centrality-based method which selects seed nodes according to node degree. Specifically, it iteratively selects a new seed node with the maximum degree. When a node is chosen, the degree of each of its neighbor will decreased (discounted) by a unit, in order to avoid influence overlap.

Our baseline methods cover a wide spectrum of methodologies, including RIS based algorithms, influence path based approach, community-based methods, and node centrality based methods. The simulation-based greedy algorithms such as Greedy, CELF or CELF++ are not considered since none of them can find the seed nodes within a rational time in the larger datasets of our experiments.

C. EVALUATION METRICS

To test the effectiveness and efficiency of our proposed approach, we utilize three measures, which were widely used in [9], [16], and [32].

Influence Spread: influence spread is defined as the expected number of overall individuals which can be successfully activated through influence propagation. It is a widely used metric to evaluate the accuracy and effectiveness of an influence maximization algorithm. Since we have previously shown that calculating the exact value of influence spread is #P-hard, an alternative way is through Monte-Carlo simulations. In this article, we calculate the influence spread by averaging over 10,000 MC simulations.

Running Time & Memory Usage: Running time and memory usage are widely used measures to evaluate the time and space efficiency of an influence maximization algorithm. In this article, we focus on the overall time and memory consumptions of different algorithms in mining $k = 50$ seed nodes and use them to as time and space measures.

D. EXPERIMENTAL PROCEDURE

1) DIFFUSION MODEL

The diffusion model used in this article is the weighted cascade (WC) model, i.e., the independent cascade model with propagation probability from u to v defined by $p_{uv} = 1/k_v$, where k_v is the degree of v.

2) EXPERIMENTAL ENVIRONMENT

Our experiments are conducted on a workstation with Intel Xeon E3 3.5GHz CPU and 32GB memory, running the Ubuntu Linux operating system. Our algorithm is programmed using C/C++ and the codes are available online.[6] The baseline methods are also programmed in C/C++, and we thank the authors for sending us their codes so that we can have a comprehensive and fair experimental comparison.

VI. RESULTS

A. INFLUENCE SPREAD

We first compare the influence spread values of different methods on the eight datasets, as shown in Figure 1, which exhibits the change of overall influence spread calculated by Monte-Carlo simulations along with k. From Figure 1, it can be easily observed that our LAIM algorithm and its fast version (FastLAIM) always outperform other baseline methods in terms of influence spread across different values of k. The path-based IPA algorithm, on the contrary, exhibits the worst overall performance. Its best performance is observed on the Epinions dataset (Figure 1(c)), while on the other datasets its performance is visibly inferior to the other baselines. TOE, as another path-based method, performs better than IPA. This is because it takes a target-oriented strategy, which provides more accurate estimation of the influence spread. However, its performance is far from satisfactory when compared with our proposed approach, as shown in Figure 1 (b), (e), (g). The node centrality based method, i.e., SD, performs well on a few networks (e.g., Amazon and Pokec). However, on most of the datasets, it is unable to provide stable solutions with high quality. For example, from the influence values on four datasets, i.e., NetHEPT, NetPHY, LiveJournal and Orkut, we can observe significant gaps of influence spread values between the SD and our LAIM/FastLAIM approaches. On the NetPHY dataset, the influence spread value ($k = 50$) of SD is 951, while that of the LAIM and FastLAIM algorithms are 1365 and 1356, respectively, about 50% improvement. The difference between our approach and the node centrality-based algorithm indicates that incorporating the diffusion parameters in influence approximation can significantly improve the quality of solutions.

[6]Source code: https://sourceforge.net/projects/linear-time-influ-max/

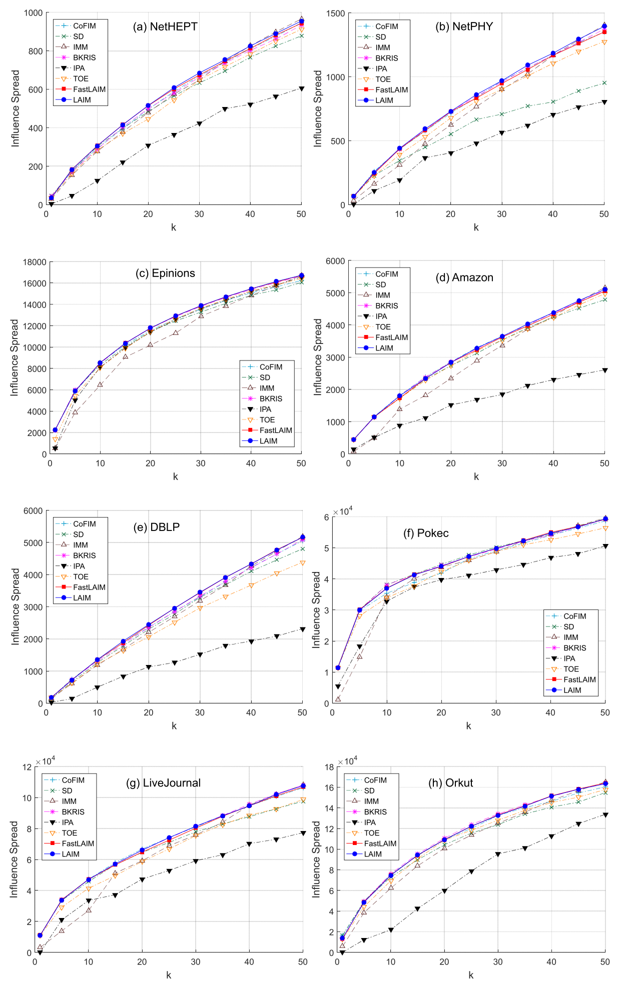


FIGURE 1. **Influence spread on eight real-world networks.**

The IMM algorithm, which is based on the RIS framework, exhibits competitive performance in terms of influence spread as compared with our LAIM and FastLAIM algorithms when k is large ($k \geq 40$), nevertheless, our approach shows its robustness across different values of k. When k is small (e.g., $k \leq 20$), on almost all of the datasets, it can be easily seen that the IMM method performs much worse than the LAIM and FastLAIM algorithms. Take the LiveJournal dataset for example, as shown in Figure 1(g), when $k = 10$, the LAIM and FastLAIM algorithms achieve influence spread values of 47,040 and 47,124 respectively, while the corresponding value of the IMM algorithm is 27,024, only slightly over half of ours.

The most competitive baselines are BKRIS and CoFIM, both were proposed very recently. The former is based on the RIS framework while the latter is based on community structure. From Figure 1, it can be observed that on most of the datasets the two algorithms and our proposed approaches cannot beat each other. However, if we take a close look at the results (e.g., Figure 1 (a), (b), (e)), we can still observe some gaps between the two baselines and our methods.

In sum, through the influence spread results on the eight datasets, our approach achieves the best overall performance and exhibits its superiority over the baseline methods in terms of both effectiveness and robustness.

B. RUNNING TIME AND MEMORY USAGE

Time efficiency is the key evaluation metric for influence maximization algorithms. Table 2 shows the running time different methods used to find $k = 50$ seeds on the eight real-world networks. Since the centrality-based algorithm (i.e., SD) naively selects k seed nodes, the comparison of its running time makes little sense, so we eliminate it from the table. From the results we see that the fast version of our linear time based approach, i.e., FastLAIM, always outperforms all the baseline algorithms in terms of time efficiency. Again, BKRIS and CoFIM are the most competitive baselines in time efficiency. Together with FastLAIM, the three methods require less than one second to find the seed nodes for all the smaller five datasets. However, on three datasets, i.e., NetHEPT, NetPHY, and Epinions, FastLAIM is about one order of magnitude faster than BKRIS and CoFIM. On the largest three datasets, i.e., Pokec, LiveJournal, and Orkut, the difference between FastLAIM and BKRIS becomes insignificant. For the other baselines, i.e, IPA, TOE, and IMM, our FastLAIM algorithm is about 1-3 orders of magnitude faster. For example, on the largest Orkut dataset with more than 3 million nodes and 100 million edges, the FastLAIM algorithm is about 28, 12, 13 times than the IPA, TOE, and IMM algorithms, respectively. The LAIM algorithm performs well on small datasets (e.g., NetHEPT, NetPHY, Epinions), but it becomes less competitive on large networks. From the largest three datasets, we see that the FastLAIM algorithm is always about 40-50 times faster than the LAIM algorithm, which is consistent with our theoretical analysis about the time complexities of the two algorithms, as shown in Theorems 4 and 5. From Table 2 we notice an interesting phenomenon that the IPA algorithm performs better on the Orkut dataset than on the LiveJournal dataset. This is mainly because IPA takes a ‘‘node-centric’’ strategy to build the influence paths, and LiveJournal has more nodes (4 million) than Orkut (3.1 million).

To deal with large-scale networks, influence maximization algorithms should also pay attention to space efficiency, as evaluated by memory usage. Table 3 exhibits the bytes of memory different methods use to find the $k = 50$ seed nodes on the eight real-world datasets. In general, we see that LAIM and FastLAIM algorithms exhibit their good scalability in handling large-scale networks. As we have shown in Theorem 6, the space complexity of our approach is linear to the network size and it requires almost no extra memory except for storing the graph. As shown in Table 3, on all the datasets, LAIM and FastLAIM require the minimum memory usage when compared with the baseline methods. For example, on the NetHEPT network, both LAIM and FastLAIM require 6M bytes of memory, only about 2.3% and 6.3% to that of the IPA and IMM methods respectively. The most competitive

TABLE 2. Running time (in seconds) of different algorithms on eight real world datasets ($k = 50$).

Method	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Pokec	LiveJournal	Orkut
FastLAIM	**0.004**	**0.01**	**0.06**	**0.2**	**0.33**	**6.98**	**9.07**	**22**
LAIM	0.08	0.86	1.96	8.6	13.5	289	498	1096
IPA	3	2	20	5	69	430	1154	605
TOE	0.16	0.35	3.1	5.53	4.44	77.8	125	273
IMM	0.85	3	4	15	15	84	83	290
BKRIS	0.07	0.22	0.27	0.89	0.80	11.4	9.2	27.1
CoFIM	0.072	0.091	0.65	0.24	0.42	8.75	13.9	73.3

TABLE 3. Memory usage of different algorithms on eight real world datasets ($k = 50$).

Method	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Pokec	LiveJournal	Orkut
FastLAIM	**6M**	**14M**	**28M**	**81M**	**94M**	**993M**	**1.8G**	**4.3G**
LAIM	**6M**	**14M**	**28M**	**81M**	**94M**	**993M**	**1.8G**	**4.3G**
IPA	263M	284M	1.2G	257M	3.7G	14.2G	31.3G	22.5G
TOE	9M	23M	49M	143M	148M	1.9G	3.3G	8.4G
IMM	95M	187M	84M	712M	642M	1.9G	2.4G	5.5G
BKRIS	18M	42M	71M	255M	247M	2.7G	5.1G	10G
CoFIM	7M	22M	47M	116M	126M	2.2G	3.5G	11G

baselines are the TOE and CoFIM methods, and their memory usages are slightly over ours on the small networks. However, on the million-scale datasets (Pokec, LiveJournal, and Orkut), our algorithms only require about half of the memory as compared to TOE and CoFIM. The IPA algorithm exhibits the worst overall performance in memory efficiency, and since it takes a "node-centric" strategy to build the influence paths, it performs better on the Orkut dataset (with less nodes) than on the LiveJournal dataset, though the former is a larger network with more edges.

In sum, the results in Table 2 and Table 3 show the extremely high time and space efficiency of our proposed approach as compared to the existing algorithms.

C. IMPACT OF ALGORITHM PARAMETER γ

As shown in subsection IV-A, our LAIM algorithm depends on an iterative parameter γ, which determines how we approximate the influence of a node and plays plays an important role for the solution quality. Figure 2 shows how the influence spread value changes as the parameter γ varies form 2 to 7 on eight real-world datasets in selecting $k = \{10, 30, 50\}$ seed nodes. As a whole, all the eight real-world datasets exhibit an growing trend of influence spread as the parameter γ increases. Moreover, the growing is more significant when γ is small, and the growth will slow down when γ continues going up. As shown in Figure 2, if the parameter γ is too small (e.g., $\gamma = 2$), the LAIM algorithm may generate unsatisfactory solutions with low influence spread. Take the DBLP dataset for example, the influence spread with respect to $k = 50$ seeds is 4,232 when $\gamma = 2$, while that value goes up to 4,849 when $\gamma = 3$, as shown in Figure 2(e). The difference becomes inconspicuous after $\gamma \geq 4$. Recalling that γ is the number of iterations in our LAIM algorithm, the results are consistent with our model assumption, i.e., the influence of a node mainly propagates within its local neighbors instead of the remote neighbors.

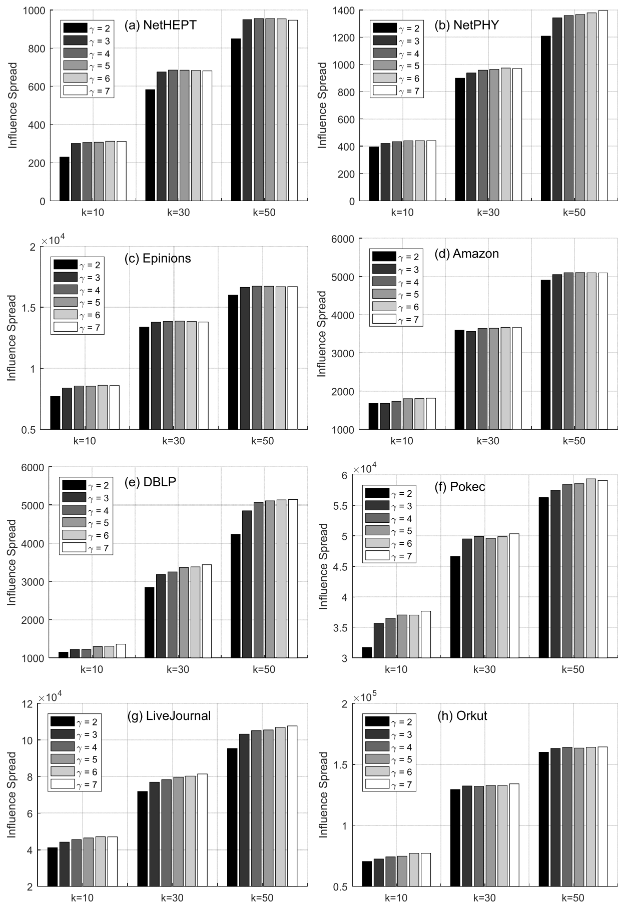


FIGURE 2. The impact of parameter γ on influence spread on eight real-world networks.

D. COMPARISON OF LAIM AND FASTLAIM

In this subsection, we further conduct experiments to compare the performance of the LAIM algorithm and its fast version, FastLAIM, in terms of both effectiveness (as evaluated

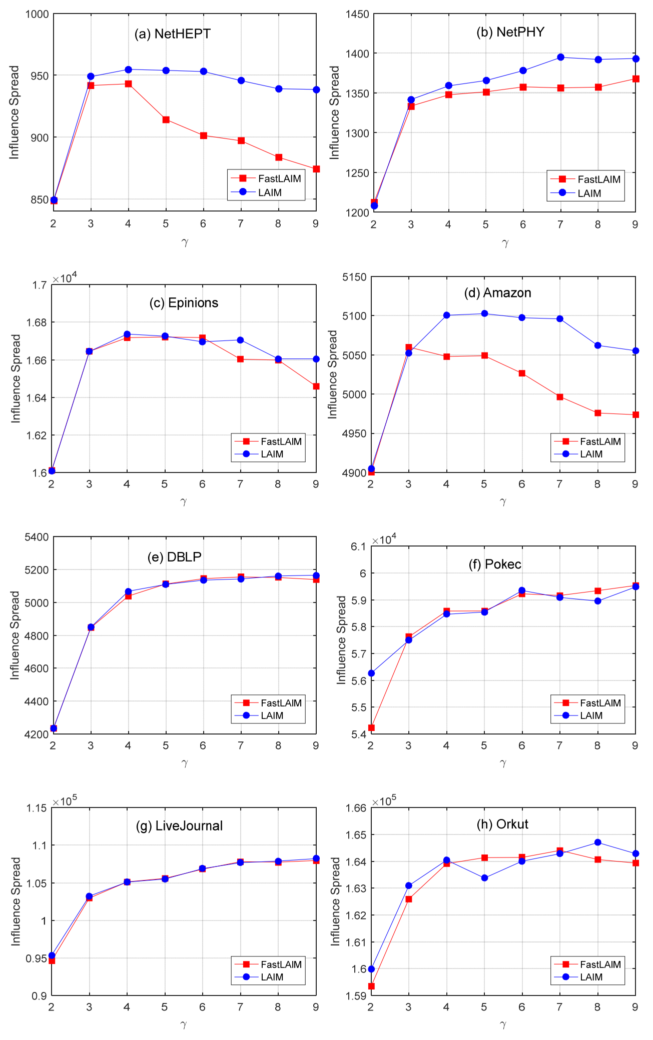


FIGURE 3. The comparison of LAIM and FastLAIM in terms of influence spread under different parameter $\gamma (k = 50)$.

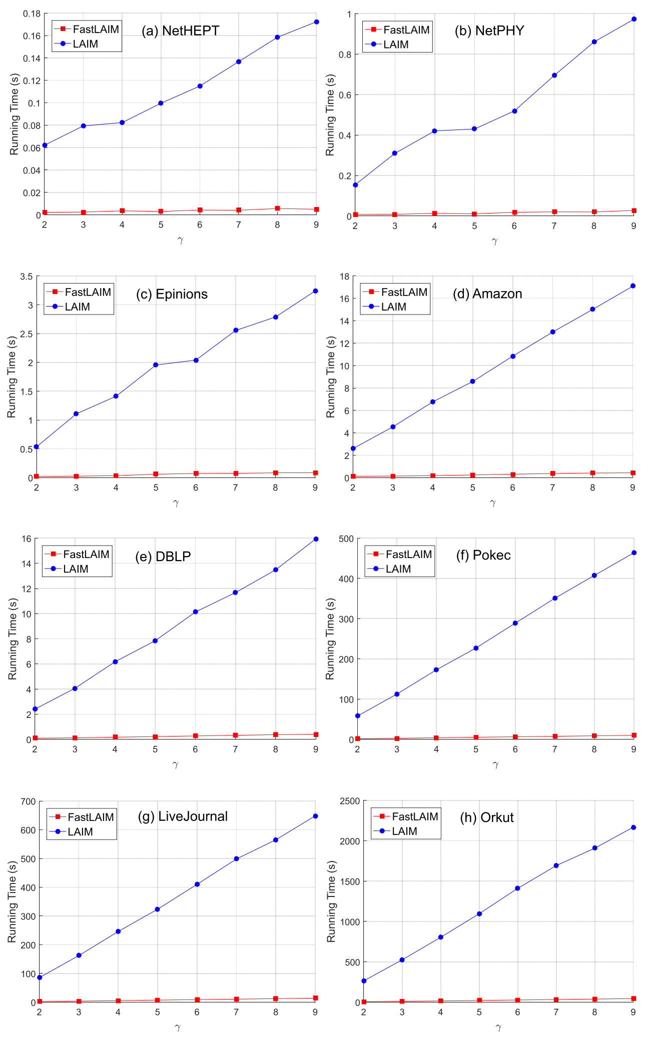


FIGURE 4. The comparison of LAIM and FastLAIM in terms of running time under different parameter γ (k=50).

by influence spread) and efficiency (as evaluated by running time).

1) COMPARISON OF INFLUENCE SPREAD

Different from the LAIM algorithm, the FastLAIM algorithm directly select the top k seed nodes with the highest $inf_G^{\leq\gamma}(v)$ values, making it much faster than the LAIM algorithm (as shown in Table 2). However, a potential drawback of this approach is that it may generate solutions of inferior quality since the influence area of different seeds may overlap with each other. We show the results in Figure 3, which exhibits how the influence spread values of the two algorithms change along with different values of γ in selecting $k = 50$ seeds. On several datasets (NetHEPT, NetPHY, and Amazon) we find that the performance of the FastLAIM algorithm becomes inferior to that of the LAIM algorithm as the value of γ increases. However, the best performances of the two algorithms are quite close to each other. In most cases, the performances of the two algorithms achieve significant improvements when γ increase from 2 to 3, and then gradually become stable as γ continues going up, which is consistent with the results in Figure 2. Based on the results of Figure 3, we can come to the conclusion that, as compared to the LAIM algorithm, the FastLAIM algorithm is able to generate high quality solutions on real-world datasets.

2) COMPARISON OF RUNNING TIME

In Theorem 4 and Theorem 5, we have proved that the time complexities of our LAIM and FastLAIM algorithms are $O(k\gamma m)$ and $O(\gamma m + n\log n)$, both of which are linear to the network size. Figure 4 shows the relationship between the running time of the two algorithms and the iterative parameter γ on eight real-world datsets. From the results we see that the FastLAIM algorithm significantly outperform the LAIM algorithm in time efficiency. The running time of the LAIM algorithm increase proportionally to the parameter γ, which is consistent with our theoretical analysis, while the FastLAIM algorithm shows its superiority in time efficiency across different values of γ. For example, on the largest Orkut dataset with 3.1 million nodes and 117 million edges, the running time of the LAIM algorithm rises from 494 seconds to 3,229 seconds as γ changes from 2 to 9, while that of the FastLAIM algorithm only grows from 13 seconds and 84 seconds, which is much more efficient.

VII. CONCLUSION

In this article, we studied the influence maximization (IM) problem under the independent cascade (IC) diffusion model, and proposed a new iterative approach which was proved to have linear time and space complexity. Our approach has two steps: (1) influence approximation; and (2) seed set selection. In the first step we proposed an iterative algorithm to compute the local influence of a node based on a recursive formula, and used the local influence to approximate its global influence. In the second step, the k influential seed nodes were mined based on the approximated influence in the first step. We conducted extensive experiments on eight real-world datasets, and showed that our proposed approach significantly outperformed existing methods in terms of both effectiveness and efficiency. Moreover, benefit from the linear time and space complexity, our approach can easily handle large-scale networks with millions of vertices and hundreds of millions of edges.

To the best of our knowledge, there are almost no influence maximization algorithms which not only generate high quality solutions, but also has linear time and space complexity. We believe our study makes significant contributions to the research of influence maximization. One limitation of our approach is that it is mainly designed for the independent cascade diffusion model. In the future, we will try to extend our approach to the linear threshold model, which also has many real-world applications.

REFERENCES

[1] S. Peng, G. Wang, and D. Xie, "Social influence analysis in social networking big data: Opportunities and challenges," *IEEE Netw.*, vol. 31, no. 1, pp. 11–17, Jan./Feb. 2017.

[2] J. J. Brown and P. H. Reingen, "Social ties and word-of-mouth referral behavior," *J. Consum. Res.*, vol. 14, no. 3, pp. 350–362, Dec. 1987.

[3] J. Goldenberg, B. Libai, and E. Müller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing Lett.*, vol. 12, no. 3, pp. 211–223, Aug. 2001.

[4] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," in *Proc. 7th ACM Conf. Electron. Commerce (EC)*, 2006, pp. 228–237.

[5] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "A data-based approach to social influence maximization," *Proc. VLDB Endowment*, vol. 5, no. 1, pp. 73–84, Sep. 2011.

[6] S. Gao, H. Pang, P. Gallinari, J. Guo, and N. Kato, "A novel embedding method for information diffusion prediction in social network big data," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2097–2105, Aug. 2017.

[7] V. Mahajan, E. Müller, and F. M. Bass, "New product diffusion models in marketing: A review and directions for research," *J. Marketing*, vol. 54, no. 1, pp. 1–26, 1990.

[8] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 137–146.

[9] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2010, pp. 1029–1038.

[10] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Math. Program.*, vol. 14, no. 1, pp. 265–294, Dec. 1978.

[11] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2009, pp. 199–208.

[12] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2007, pp. 420–429.

[13] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "CELF++: Optimizing the greedy algorithm for influence maximization in social networks," in *Proc. 20th Int. Conf. Companion World Wide Web*, 2011, pp. 47–48.

[14] Q. Liu, B. Xiang, E. Chen, H. Xiong, F. Tang, and J. X. Yu, "Influence maximization over large-scale social networks: A bounded linear approach," in *Proc. 23rd ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 171–180.

[15] H.-L. Liu, C. Ma, B.-B. Xiang, M. Tang, and H.-F. Zhang, "Identifying multiple influential spreaders based on generalized closeness centrality," *Phys. A Stat. Mech. Appl.*, vol. 492, pp. 2237–2248, Feb. 2018.

[16] J. Kim, S.-K. Kim, and H. Yu, "Scalable and parallelizable processing of influence maximization for large-scale social networks?" in *Proc. IEEE 29th Int. Conf. Data Eng.*, Apr. 2013, pp. 266–277.

[17] B. Liu, G. Cong, Y. Zeng, D. Xu, and Y. M. Chee, "Influence spreading path and its application to the time constrained social influence maximization problem and beyond," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1904–1917, Aug. 2014.

[18] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 75–86.

[19] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2015, pp. 1539–1554.

[20] H. Wu, J. Shang, S. Zhou, and Y. Feng, "A linear time algorithm for influence maximization in large-scale social networks," in *Proc. Int. Conf. Neural Inf. Process.*, Oct. 2017, pp. 752–761.

[21] C. Zhou, P. Zhang, W. Zang, and L. Guo, "On the upper bounds of spread for greedy algorithms in social network influence maximization," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 10, pp. 2770–2783, Oct. 2015.

[22] J. Zhu, Y. Liu, and X. Yin, "A new structure-hole-based algorithm for influence maximization in large online social networks," *IEEE Access*, vol. 5, pp. 23405–23412, 2017.

[23] F. Riquelme, P. Gonzalez-Cantergiani, X. Molinero, and M. Serna, "Centrality measure in social networks based on linear threshold model," *Knowl.-Based Syst.*, vol. 140, pp. 92–102, Jan. 2018.

[24] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 88–97.

[25] Y.-Y. Ko, D.-K. Chae, and S.-W. Kim, "Accurate path-based methods for influence maximization in social networks," in *Proc. Int. Conf. World Wide Web*, 2016, pp. 59–60.

[26] Y.-Y. Ko, D.-K. Chae, and S.-W. Kim, "Influence maximisation in social networks: A target-oriented estimation," *J. Inf. Sci.*, Feb. 2018.

[27] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2014, pp. 946–957.

[28] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. Int. Conf. Manage. Data*, 2016, pp. 695–710.

[29] K. Huang, S. Wang, G. Bevilacqua, X. Xiao, and L. V. S. Lakshmanan, "Revisiting the stop-and-stare algorithms for influence maximization," *Proc. VLDB Endowment*, vol. 10, no. 9, pp. 913–924, May 2017.

[30] X. Wang, Y. Zhang, W. Zhang, X. Lin, and C. Chen, "Bring order into the samples: A novel scalable method for influence maximization," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 2, pp. 243–256, Feb. 2017.

[31] J. Shang, H. Wu, S. Zhou, L. Liu, and H. Tang, "Effective influence maximization based on the combination of multiple selectors," in *Proc. Int. Conf. Wireless Algorithms Syst. Appl.*, 2017, pp. 572–583.

[32] J. Shang, S. Zhou, X. Li, L. Liu, and H. Wu, "CoFIM: A community-based framework for influence maximization on large-scale networks," *Knowl.-Based Syst.*, vol. 117, pp. 88–100, Feb. 2017.

[33] J. Yang and J. Liu, "Influence maximization-cost minimization in social networks based on a multiobjective discrete particle swarm optimization algorithm," *IEEE Access*, vol. 6, pp. 2320–2329, 2017.

[34] H. Li, L. Pan, and P. Wu, "Dominated competitive influence maximization with time-critical and time-delayed diffusion in social networks," *J. Comput. Sci.*, Oct. 2017.
[35] X. Wang, Y. Zhang, W. Zhang, and X. Lin, "Efficient distance-aware influence maximization in geo-social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 3, pp. 599–612, Mar. 2017.
[36] X. Li, X. Cheng, S. Su, and C. Sun, "Community-based seeds selection algorithm for location aware influence maximization," *Neurocomputing*, vol. 275, pp. 1601–1613, Jan. 2018.
[37] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic Web," in *Proc. 2nd Int. Semantic Web Conf.*, vol. 2870, 2003, pp. 351–368.
[38] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, 2015.
[39] L. Takac and M. Zabovsky, "Data analysis in public social networks," in *Proc. Int. Sci. Conf. Int. Workshop Present Day Trends Innov.*, vol. 1, no. 6, May 2012.

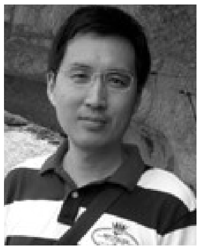
HONGCHUN WU was born in Heze, China, in 1992. He received the B.S. degree in information and computing science from the China University of Geosciences in 2015. He is currently pursuing the M.S. degree in computer science and technology with Chongqing University. He is currently with the Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing University. He has authored multiple high-quality journal and conference articles, including KBS, WASA, and ICONIP. His research interests include machine learning, data mining, social networks analysis, and recommender systems. He was a recipient of many scholarships.

JIAXING SHANG was born in Guiyang, China, in 1987. He received the B.S. and Ph.D. degrees in control science and engineering from Tsinghua University, Beijing, China, in 2010 and 2016, respectively. Since 2016, he has been a Post-Doctoral Researcher of computer science and technology with Chongqing University, Chongqing, China, where he is currently a Lecturer with the College of Computer Science. He has authored about 20 high-quality journal and conference articles, including KBS, WASA, ICONIP, SNA-KDD, and *Physica A*. His research interests include social networks analysis, data mining, artificial intelligence, and recommender systems.

SHANGBO ZHOU was born in Guangxi, China. He received the B.S. degree in mathematics from the Guangxi National College in 1985, the M.S. degree in mathematics from Sichuan University in 1991, and the Ph.D. degree in circuit and system from Electronic Science and Technology University in 2003. From 1991 to 2000, he was with the Chongqing Aerospace Electronic and Mechanical Technology Design Research Institute. Since 2003, he has been with the College of Computer Science, Chongqing University, where he is currently a Professor. He has authored over 100 journal and conference papers, including *Physical Review E*, *Neurocomputing*, *Chaos*, *Pattern Recognition*, and *Multimedia Tools and Applications*. His current research interests include artificial neural networks, physical engineering simulation, visual object tracking, and nonlinear dynamical system.

YONG FENG was born in Chongqing, China, in 1977. He received the B.S. degree in computer applied technology, the M.S. degree in computer systems organization, and the Ph.D. degree in computer software and theory from Chongqing University, China, in 1999, 2003, and 2006, respectively.

From 2007 to 2010, he was a Post-Doctoral Researcher with the Control Science and Engineering Center, Chongqing University, where he is currently a Professor with the College of Computer Science. He has authored over 50 academic papers and two monographs.



BAOHUA QIANG was born in Nanyang, China, in 1972. He received the B.S. and M.S. degrees from Southwest University in 1996 and 2002, respectively, and the Ph.D. degree from Chongqing University in 2005. In 2007, he joined the University of Illinois as a Visiting Scholar. From 2007 to 2009, he was a post-doctor with the South China University of Technology. He is currently a Professor with the Guilin University of Electronic Technology. He has authored one book, over 60 articles, and over 10 inventions. His major research interests include Web information processing, intelligent search, massive data processing, and network information integration.

WU XIE was born in Yichun, China, in 1979. He received the B.S. and M.S. degrees from the Guilin University of Electronic Technology in 2002 and 2005, respectively, and the Ph.D. degree from Xidian University in 2015. He is currently an Associate Professor with the Guilin University of Electronic Technology. He has authored over 20 articles. His research interests include data mining of social networks, big data, and cloud computing.