Influential Node Tracking on Dynamic Social Network: An Interchange Greedy Approach

Guojie Song, Yuanhao Li, Xiaodong Chen, Xinran He and Jie Tang

Abstract—As both social network structure and strength of influence between individuals evolve constantly, it requires to track the influential nodes under a dynamic setting. To address this problem, we explore the Influential Node Tracking (INT) problem as an extension to the traditional Influence Maximization problem (IM) under dynamic social networks. While Influence Maximization problem aims at identifying a set of k nodes to maximize the joint influence under one static network, INT problem focuses on tracking a set of influential nodes that keeps maximizing the influence as the network evolves. Utilizing the smoothness of the evolution of the network structure, we propose an efficient algorithm, Upper Bound Interchange Greedy (UBI) and a variant, UBI+. Instead of constructing the seed set from the ground, we start from the influential seed set we find previously and implement node replacement to improve the influence coverage. Furthermore, by using a fast update method by calculating the marginal gain of nodes, our algorithm can scale to dynamic social networks with millions of nodes. Empirical experiments on three real large-scale dynamic social networks show that our UBI and its variants, UBI+ achieves better performance in terms of both influence coverage and running time.

Index Terms—Influence Maximization, Influential Nodes Tracking, Social Network, Scalable Algorithm.

1 Introduction

The processes and dynamics by which information and behaviors spread through social networks have long interested scientists within many areas. Understanding such processes have the potential to shed light on the human social structure, and to impact the strategies used to promote behaviors or products. While the interest in the subject is long-standing, recent increased availability of social network and information diffusion data (through sites such as Facebook, Twitter, and LinkedIn) has raised the prospect of applying social network analysis at a large scale to positive effect.

One particular application that has been receiving interest in enterprises is to use word-of-mouth effects as a tool for viral marketing. Motivated by the marketing goal, mathematical formalizations of influence maximization have been proposed and extensively studied by many researchers [1], [2], [3], [4], [5], [6], [7], [8], [9]. Influence maximization is the problem of selecting a small set of seed nodes in a social network, such that their overall influence on other nodes in the network, defined according to particular models of diffusion, is maximized.

Marketing campaign is usually not a one-time deal, instead enterprises carry out a sustaining campaign to promote their products by seeding influential nodes continuously. Often, a marketing campaign may last for months or years, where the company periodically allocates budgets to the selected influential users to utilize the power of the word-of-mouth effect. Under this situation, it is natural and important to realize that social or information networks are always dynamics, and their topology evolves constantly over time [10], [11], [12]. For example, links appear and disappear when users follow/unfollow others in Twitter or friend/unfriend others in Facebook. Moreover, the strength of influence also keeps changing, as you are more influenced by your friends who you contact frequently, while the influence from a friend usually dies down as time elapses if you do not contact with each other. As a result, a set of nodes influential at one time may lead to poor influence coverage after the evolution of social network, which suggests that using one static set as seeds across time could lead to unsatisfactory performance.

It turns out that targeting at different nodes at different time becomes essential for the success of viral marketing. We proceed to illustrate the idea of considering the dynamic perspect in influence maximization using an example in Figure 1. In this example, users are connected by edges at different time, each of which indicates a user may influence over another user. Numbers over each edge give the corresponding influencing probabilities. For example, there is an edge between v1 and v3 at $t = 0$ and the edge is deleted at $t = 1$. And user v1 will influence v2 with a probability of 0.7 at $t = 0$, and the influencing probability is 0.2 at $t = 1$. This means that user v1 would no longer influence v3 at $t = 1$ and v2 cannot be activated by v1 by probability 0.7 at $t = 1$. Suppose we are asked to find a single seed user to maximize the expected number of influenced users. Without any dynamic constraint, that is all the snapshots are aggregated into one weighted static graph, user v1 will be returned as the result. Intuitively, it is expected to influence

- *Guojie Song is with Key Laboratory of Machine Perception, Peking University, Beijing. E-mail: gjsong@pku.edu.cn*
- *Yuanhao Li is with Key Laboratory of Machine Perception, Peking University, Beijing. E-mail: liyh98@126.com*
- *Xiaodong Chen is with Key Laboratory of Machine Perception, Peking University, Beijing. E-mail: chenxd@pku.edu.cn*
- *Xinran He is with University of Southern California, USA. E-mail: xinranhe@usc.edu*
- *Jie Tang is with Department of Computer Science and Technology, Tsinghua University, Beijing. E-mail: jietang@tsinghua.edu.cn*

the maximal number of users among all users. However, if we aim to find a single seed user that influences the maximal number of users at different time, user v2 will become the new result at time $t = 1$. Intuitively, this is because v1 can at most influence v4 at $t = 1$ while v2 influences v1, v3 and v4 with a higher probability as given in Figure 1.

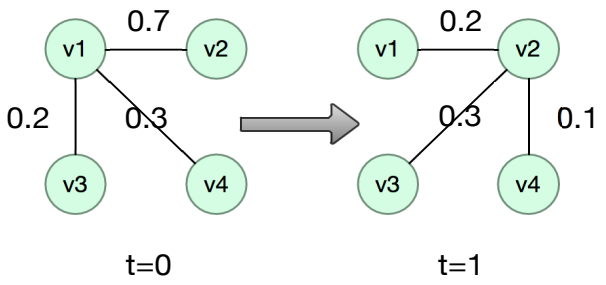


Fig. 1. An example illustrating the influence maximization with dynamic prospect

However,traditional algorithms for Influence Maximization become inefficient under this situation as they fail to consider the connection between social networks at different time and have to solve many Influence Maximization problems independently for social network at each time. In this paper, we propose an efficient algorithm, *Upper Bound Interchange Greedy* (UBI), to tackle Influence Maximization problem under dynamic social network, which we term as *Influential Node Tracking* (INT) problem. That is to track a set of influential nodes which maximize the influence under the social network at any time.

The main idea of our UBI algorithm is to leverage the similarity of social networks near in time and directly discover the influential nodes based on the seed set found for previous social network instead of constructing the solution from an empty set. As similarity in network structure leads to similar set of nodes that maximize the influence. In our UBI algorithm, we start from the seed set maximizing the influence under previous social network. Then we change the nodes in the existing set one by one in order to increase the influence under the current social network. As the optimal seed set differs only in a small number of nodes, a few rounds of node exchanges are enough to discover a seed set with large joint influence under current social network. Moreover, it can be shown that the above node exchange procedure leads to a constant approximation guarantee of $1/2$, when certain stopping criteria is applied to node exchanges.

Our method requires a large number of computations in evaluating the node replacing gain, which takes unaffordable long time if traditional Monte-Carlo simulations are applied. In order to scale our algorithm up to large networks, we utilize the Upper Bound Based Approach proposed by Zhou et al. to reduce the calls of Monte-Carlo simulations [13]. We first tighten their bound by excluding all the influence paths with edges into the seed set and most important we design an efficient method to update the upper bound as the underlying network structure changes instead of carrying out expensive matrix operation for each individual network, as the result, we propose UBI and its variant, UBI+.

Extensive experiments are conducted on three real dynamic networks of different types and scales. The comparison of our method to several state-of-arts Influence Maximization algorithms for static network shows that our methods leads to both larger influence coverage and less running time. We show that our UBI algorithm achieves comparable influence coverage as Greedy algorithm within only seconds for networks with millions of nodes across multiple snapshots. Also, the variant algorithm, UBI+ are conducted on the same networks and show better performance than UBI.

Our contributions can be summarized as follows:

- We explore the Influential Node Tracking (INT) problem as an extension to the traditional Influence Maximization problem to maximize the influence coverage under a dynamic social network.
- We propose an efficient algorithm, Upper Bound Interchange (UBI) to solve the INT problem. Our algorithm achieves comparable results as hill-climbing greedy algorithm where the $1 - 1/e$ approximation is guaranteed. The algorithm has the time complexity of O(kn), and the space complexity of O(n), where n is the number of nodes and k is the size of the seed set.
- We propose an algorithm UBI+, based on UBI, that improves the computation of node replacement upper bound.
- We evaluate the performance on large-scale real social network. The experiment results confirm our theoretical findings and show that our UBI and UBI+ algorithm achieve better performance of both influence coverage and running time.

Paper Organization. We summarize the related literatures in section 2. In section 3, we formally formulate our Influential Node Tracking problem after introducing the diffusion model and the Influence Maximization problem. We then present our efficient UBI algorithm and its variant, UBI+ algorithm for the INT problem in section 4. In section 5, we present our experiment results on three real-world large-scale dynamic social networks and we conclude our work with discussion on future work in section 6.

2 RELATED WORK

Domingos et al. [2], [14] first study the influence maximization problem, while Kempe et al. [3] later establish the problem formally as a discrete optimization problem and propose a hill-climbing greedy algorithm with a $1 - 1/e$ approximation guarantee. However, the proposed solution does not scale to large networks as it requires a large number of Monte-Carlo simulations for influence estimation.

Following the seminal work [3], many researchers have been working on design efficient algorithms for Influence Maximization problem, leading to a large number of different methods [1], [4], [5], [6], [7]. The proposed methods can be mainly categorized into two types. The first type

of algorithms aims at improving the efficiency of the hill-climbing greedy algorithm while preserving the $1 - 1/e$ approximation guarantee [13], [15]. For example, Leskovec et al. design the CELF method to accelerate the greedy algorithm by utilizing the sub modularity of the objective function to carry out lazy evaluation [15]. More recently, Zhou et al. have achieved further acceleration by incorporating upper bound on the influence function [13]. Based on the idea that $p_{u,v}^{G(S)} <= p_{u,v}^{G}$, in this work, we utilize the same idea in our UBI algorithm with an improved upper bound for node replacement gain. Moreover, we extract the formula that is used to calculate the node replacement gain to two parts of marginal gain and then our major task becomes to provide an upper bound and a lower bound of the marginal gain. With the calculation of the upper and the lower bound on the terms, we achieve a much tighter bound than just improving the method of [13]. Moreover, we design an efficient method to update the upper bound as network structure changes.

On the other hand, the second type of algorithms applies various heuristics without provable approximation guarantee [1], [4], [5], [6], [16], [17], [18], [19], [20]. For instances, Jung et al. proposes the state-or-art algorithm IRIE for Influence Maximization problem based on the idea of PageRank. While Jiang et al. use simulated annealing to optimize the influence function [16] while Wang et al. utilize community structure to accelerate influential node discovery [5].

However, all the previous methods aim to discover the influential nodes under one static network. As far as we are concerned, the only paper on Influence Maximization under dynamic networks is by Aggarwal et al. [21]. Nevertheless, their work is merely marginally related to this paper in that they focus on finding a seed set at time t, that maximizes the influence at some $t + \Delta$ given the dynamics of the evolution of network during the interval $[t, t + \Delta]$. In contrast, in our work, we consider fast update of seed set across different snapshot graphs, each of which is a static network that we would like to maximize the influence of the seed set. The major difference is that in their work, the diffusion process is taking place under a dynamic network while we consider maximizing the influence under a series of static snapshots taking from a dynamic social network. Zhuang et al. [22] study the influence maximization under dynamic networks where the changes can be only detected by periodically probing some nodes. Their goal then is to probe a subset of nodes in a social network so that the actual influence diffusion process in the network can be best uncovered with the probing nodes. That means, their algorithm is to minimize the possible error between the observed network and the real network through probing a small portion of the network. In contrast, the whole structure of the dynamic network is known and our goal is to track the influential nodes and try to maximize the influence coverage of a particular size of seed set. We focus on fast tracking of influential nodes. Moreover, our algorithm can be applied when the changes in network structure have already been discovered by their probing method.

3 PRELIMINARIES AND PROBLEM STATEMENT

In this section, we first introduce the diffusion model, namely the *Independent Cascade Model* and the *Influence Maximization* for static network. We then formally state our *Influential Node Tracking* problem as a generalization of the Influence Maximization problem to dynamic social networks. Table 1 lists the symbol notations used in this paper.

TABLE 1
Notations

Notations	Descriptions
$\mathcal{G} = \{G^t\}_1^T$	a dynamic social network
$G^t = (V^t, E^t)$	a snap shot of $\mathcal{G}$ at time t
V^t	the vertex set of G^t
E^t	the edge multiset of G^t
$p_{u,v}^G$	the strength of influence of nodes u on v in snapshot G
$G(T)$	the subgraph of G by excluding the edges associated to nodes in T
$\delta_{v,v_s}(S)$	the replacing gain of changing from v_s to v
$\bar{\delta}_{v,v_s}(S)$	the upper bound of the replacing gain $\delta_{v,v_s}(S)$
S^t	the seed set at time t
k	the size of seed set
$\sigma(S)$	the expectation of nodes influenced by S
$\rho_S(T)$	the marginal gain of interchanging by adding set S to the existing node set T
$U(T)$	the upper bound column vector of the marginal gain of nodes set T
$L(T)$	the lower bound column vector of the marginal gain of nodes set T
$AP_{v,i}(S)$	the probability that v is activated exactly at step i under the seed set S
$AP_{v,i}(S\|T)$	the probability that node v is activated exactly at step i without the help from nodes in set T

3.1 Diffusion Model and the Influence Maximization Problem

In this work, we study the social influence under the widely adopted Independent Cascade (IC) model. Under the IC model, the social network is modeled as a directed network $G = (V, E)$, where V corresponds to the individuals while E represents the sets of social links between the individuals. Moreover, each edge $(u, v) \in E$ is associated with a propagation probability $p_{u,v}^G$ indicating the strength of influence of individual u on v. When G is clear from the context, we simply use $p_{u,v}$ to keep the notations uncluttered.

The IC model describes a simple and intuitive diffusion process. Starting from a seed set S, which begins active (having adopted the behavior), the diffusion process unfolds in discrete time steps as follows. When a node u becomes active in step t, it attempts to activate all currently inactive neighbors in step $t+1$. For each neighbor v, it succeeds with the known probability $p_{u,v}$. If it succeeds, v becomes active; otherwise, v remains inactive. Once u has made all these attempts, it does not get to make further activation attempts at later times.

Given the seed set S, we define the influence coverage of S as the expected number of activated nodes when the diffusion process ends, denoted by the influence function $\sigma(S)$. The *Influence Maximization* (IM) problem under the IC model aims at finding a seed set $S \subseteq V$ of size at most k to maximize the influence function $\sigma(S)$. Formally, the IM problem is defined as the following optimization problem:

$$S^* = \arg\max_{|S|\leq k} \sigma(S)$$

Though it has been shown by Kempe et al. in [3] that the IM problem under IC model is NP-hard, the following good properties of the IC model allow for approximate algorithm to discover the influential nodes: the influence function $\sigma(S)$ under the IC model is monotone and submodular [3]. [1]

The above properties lead to a simple greedy algorithm (Algorithm 1) proposed by Nemhauser et al. [23] for maximizing monotone submodular functions. The algorithm repeatedly chooses the node with the maximum marginal gain and adds it to the current seed set until the budget k is reached. Proved by [23], this algorithm approximates the optimal solution with a factor of the $(1 - 1/e)$ for the Influence Maximization problem.

Algorithm 1 Greedy($G = (V, E)$, k)

1: initialize $S = \emptyset$
2: **for** $i = 1$ to k **do**
3: $\quad v^* = \arg\max_{v \in V - S} \{\sigma(S + \{v\}) - \sigma(S)\}$
4: $\quad S = S + \{v^*\}$
5: **end for**
6: **Output** S

However, the exact computation of the marginal gain has shown to be #P-hard in [6], though approximate estimation can be achieved via multiple times of Monte-Carlo simulations, which are extremely inefficient for large networks. To tackle the inefficiency of the above greedy algorithm, numerous methods are proposed, for example [1], [6], [13], [15], [17]. Though with much better efficiency, the algorithms may still spend at least minutes on a network with millions of nodes.

3.2 Influential Node Tracking Problem

The traditional Influence Maximization problem aims at finding influential nodes for only one static social network. However, real-world social networks are seldom static. Both the structure and also the influence strength associated with the edges change constantly. As a result, the seed set that maximizes the influence coverage should be constantly updated according to the evolution of the network structure and the influence strength.

In this work, we model the dynamic social network as a series of snapshot graphs, G_1,..., G_T. We assume that the nodes remain the same while the edges in each snapshot graph change across different time intervals. Each snapshot graph is modeled as a directed network, $G^t = (V, E^t)$, which includes edges appearing during the periods under consideration. Moreover, a set of propagation probabilities $P^t_{u,v}$ is associated with each snapshot graph G^t.

Our goal is to track a series of seed sets, denoted as $S^t, t = 1, \ldots, T$, that maximizes the influence function $\sigma^t(\cdot)$ at each of the snapshot G^t. More formally, we define the above task as the Influential Node Tracking problem.

1. Recall that a set function f is monotone if $f(S+x) \geq f(S)$ for any element x; and f is submodular if it has diminishing returns: $f(S + x) - f(S) \geq f(T + x) - f(T)$ for any element x whenever $S \subseteq T$.

Influential Nodes Tracking (INT). Let $\mathcal{G} = \{G^t\}_1^T$ be a dynamic social network. The influential nodes tracking problem is to discover a series of seed sets $S^1, \ldots, S^T$ whose size is at most k, such that $S^t = \arg\max_{S \in V, |S| \leq k} \sigma^t(S)$ for all snapshot graphs $G^t, t = 1, \ldots, T$.

The most naive and straight-forward way to solve the INT problem is to treat the different snapshot graphs independent and solve them as separate Influence Maximization problem for each snapshot G^t by algorithms such as [6], [13], [15], [17].

However, as solving Influence Maximization problem for a single graph with moderate size already costs several minutes, the running time of computing influence nodes for a large set of graphs becomes unaffordable. Moreover, aiming at tracking influential nodes in real time, we do need an efficient algorithm to discover the influential nodes in short period of time. In next section, we will show how we propose a new method UBI to solve the INT problem efficiently.

4 Proposed Methods

For real dynamic social network, it is unlikely to have abrupt and drastic changes in graph structure in a short period of time. As a result, the similarity in structure of graphs from two consecutive snapshots could lead to similar seed sets that maximize the influence under each graph.

Based on the above idea, we propose UBI algorithm for the INT problem, in which we find the seed set that maximizes the influence under G^{t+1} based on the seed set S^t we have already found for graph G^t. Instead of constructing the seed set for graph G^{t+1} from the ground, we start with S^t and continually update by replacing the nodes in S^t to improve the influence coverage. Our algorithm first uses an initial set and several rounds of interchange heuristic to maximize the influence, as mentioned in the paper. So the interchange heuristic obviously works on a snapshot graph. When extended to the dynamic graph, our algorithm only needs to interchange for a few more rounds after each time window and can achieve a faster update. More detailed descriptions about how our method works on the snapshot graphs and dynamic networks will be presented in the next two subsections.

4.1 Interchange Heuristic

We use the *Interchange Heuristic* proposed in [23] as our strategy to replace the nodes in S^t. Starting from an arbitrary set $S \subseteq V$, Interchange Heuristic means to find a subset $S' \subseteq V$ that differs from S by one node and has the same cardinality.

It has been shown by Nemhauser et al. in [23] that applying Interchange Heuristic to monotone submodular function until no further improvement is possible leads to a solution with approximation guarantee $1/2$.

However, it remains to specify how we should choose set S' in the Interchange Heuristic. In this work, we choose S' in order to maximize the gain achieved via the replacement for any fixed $v_s \in S$. Let $\delta_{v,v_s}(S)$ be the replacing gain by changing from $v_s \in S$ to $v \in V - S$. Let $v^* = \arg\max_v \delta_{v,v_s}(S)$, we choose $S' = S - v_s + v^*$.

This strategy needs to evaluate the gain by replacing v_s with any node in $v \in V - S$, which calls for $|V - S|$ times of influence estimation. The calculation by running Monte-Carlo simulations is unaffordable even for network with moderate size. Inspired by the UBLF optimization proposed in [13], we use the upper bound on replacing gain to reduce a large number of influence estimations.

Assume that we have already calculated the upper bound on replacing gain $\delta_{v,v_s}(S)$ for any node $v \in V-S$. Let the upper bound on the replacement gain be $\bar{\delta}_{u,v_s}(S)$, then if for node u such that $\bar{\delta}_{u,v_s}(S) \leq \delta_{v,v_s}(S)$, the expensive computation of replacing gain for node u becomes unnecessary as its gain is guaranteed to be less or equal than that for node v. The computation of $\bar{\delta}_{u,v_s}(S)$ will be presented in next section.

We use the subroutine in Algorithm 2 to carry out the Interchange Heuristic for any fixed $v_s \in S$. If the largest replacing gain δ_{v,v_s} is less than a given threshold with $\gamma \geq 0$, we stop to find another v_s for interchange (line 5-7). This reduces the computations for the case of insignificant improvements and accelerates the process of interchange. It remains to show how v_s is selected to complete the description of our method. It turns out that we utilize the derived bounds to choose v_s with the largest possible replacing gain, namely, $v_s = \arg\max_{v_s \in S}\{\max_{v \in V-S} \bar{\delta}_{v,v_s}(S)\}$.

Algorithm 2 Interchange($G = (V, E), S, v_s, \bar{\delta}_{\cdot,v_s}(S)$)

1: **Set** $\delta_{v,v_s} \leftarrow \bar{\delta}_{v,v_s}(S)$**,** $v \in V - S$
2: **Set** $cur_v \leftarrow$ **false,** $v \in V - S$
3: **while true do**
4: $v^* = \arg\max_{v \in V-S} \{\delta_{v,v_s}\}$
5: **if** $\delta_{v^*,v_s} \leq \gamma\sigma(S)$ **then**
6: **break**
7: **end if**
8: **if** cur_{v^*} **then**
9: $S \leftarrow S - v_s + v^*$
10: **break**
11: **else**
12: $\delta_{v^*,v_s} \leftarrow \sigma(S - v_s + v^*) - \sigma(S)$
13: $cur_{v^*} \leftarrow$ **true**
14: **end if**
15: **end while**
16: **Output** S

With the interchange strategy defined above, we present our Upper Bound Interchange Greedy, in short UBI as Algorithm 3.

Algorithm 3 UBI($G = (V, E), S$)

1: Compute $\bar{\delta}_{v,v_s}(S)$ for $v \in V - S, v_s \in S$
2: **for** $i = 1$ to $|S|$ **do**
3: $v_s^* = \arg\max_{v_s \in S}\{\bar{\delta}_{\cdot,v_s}(S)\}$
4: $S \leftarrow$Interchange($G, S, v_s^*, \bar{\delta}_{\cdot,v_s}(S)$)
5: Update $\bar{\delta}_{v,v_s}(S)$ for any $v \in V - S, v_s \in S$ according to the interchange result
6: **end for**
7: **Output** S

It should be noticed that instead of carrying out node replacement until no further improvement is possible, we apply at most $|S|$ rounds of replacement in our implementation. While sacrificing the theoretical guarantee, we significantly improve the efficiency of our method, as it may take an exponential number of replacements until no improvement exists. As we will illustrate in the empirical experiments, the proposed method achieves comparable results as the hill-climbing greedy algorithm where the $1-1/e$ approximation is guaranteed.

Time and space complexities Let n (resp. m) be the number of nodes (resp. edges) in social network G. The first lines of Algorithm 3 take O(n) time. For the entire for loop, the dominant cost is on interchanging the nodes in seed set. In the worst case, the algorithm 2 needs O(n) to explore all nodes in the graph. Thus the running time is O(kn) for the for loop, which is also the time complexity of Algorithm 3. In addition to the input social graph, Algorithm 2 only needs to store bounds and replacement gain for each node, the space needed by which is O(n). Thus the space complexity of Algorithm 2 is O(n+m), which is dominated by the input of social network.

4.2 Upper Bound of Node Replacement Gain

In this section, we illustrate the only mysterious part in our UBI algorithm, namely the computation of the upper bound of the replacement gain $\bar{\delta}_{u,v_s}(S)$. Zhou et al. first use the upper bound on influence function to accelerate the greedy algorithm in influential seeds selection [13]. Following their methodology, we propose a tighter upper bound on the replacement gain by excluding the influence along paths, which include incoming edges to the seed set.

Basically, our task is to compute an upper bound on $\delta_{v,v_s}(S)$ for any $v \in V - S$ in order to accelerate the Interchange Heuristic subroutine. We have

$$\begin{aligned} \delta_{v,v_s}(S) &= \sigma(S - v_s + v) - \sigma(S) \\ &= \rho_v(S - v_s) - \rho_{v_s}(S - v_s) \end{aligned} \quad (1)$$

where $\rho_S(T) = \sigma(S + T) - \sigma(T)$ is the marginal gain by adding set S to the existing node set T. The major task is to provide an upper bound on the first term $\rho_v(S - v_s)$ and a lower bound on the second term $\rho_{v_s}(S - v_s)$. In the next two sections we will provide the upper bound and the lower bound of the marginal gain.

4.2.1 Upper Bound of Marginal gain

In this section, we illustrate the computation of the upper bound on the marginal gain $\rho_S(T)$. Let $AP_{v,i}(S)$ be the probability that node v is activated exactly at step i under the seed set S. The essential step to achieve a tighter bound is to use probability, $AP_{v,i}(S|T)$ instead of $AP_{v,i}(S)$ used in [13]. Informally, $AP_{v,i}(S|T)$ stands for the probability that node v is activated exactly at step i without the help from nodes in set T. Let $G(T)$ be the graph where the set of node T is "excluded" from G in terms of the diffusion process, namely the propagation probability $p_{u,v}^{G(T)}$ associated with $G(T)$ is defined as follows:

$$p_{u,v}^{G(T)} = \begin{cases} 0 & v \in T \\ p_{u,v}^{G} & \text{otherwise} \end{cases}$$

Then, $AP_{v,i}(S|T)$ can be formally defined as the probability that node v is activated exactly at step i under the modified graph $G(T)$. It should be noticed that $AP_{v,i}(S|T) = AP_{v,i}(S|S+T)$ as nodes in S are already activated at the beginning, thus removing the incoming edges to nodes in set S does not matter.

We need the next lemma to characterize the properties of $AP_{v,i}(S|T)$ in order to derive our bound for replacement gain.

Lemma 1. For any $v \in V$, $S,T \subseteq V$,$S \cap T = \emptyset$ and $i = 0, 1, ..., |V-S|$, we have:

$$AP_{v,i}(S+T) - AP_{v,i}(T) \leq AP_{v,i}(S|T) = AP_{v,i}(S|S+T)$$

Proof: By results in [3], the set of active nodes at each step under the IC model can be characterized alternatively as follows: for each ordered pair (u,v) independently, insert the directed edge (u,v) with probability $p_{u,v}$. Then, the active nodes at step i are exactly the ones with distance i from the seed set S. We prove the lemma by coupling the following three diffusion process with the same random choices X on the edge insertion: (1) diffusion process under graph G with seed set $S+T$; (2) diffusion process under graph G with seed set T and (3) diffusion process under graph $G(T)$ with seed set S. We abuse the notation a little bit to use $AP^X_{v,i}(S+T)$, $AP^X_{v,i}(T)$ and $AP^X_{v,i}(S|T)$ as an indicator function that takes value 1 if the node v is activated at step i in diffusion process (1), (2) and (3) respectively under random edge insertion result X and value 0 otherwise. We have $AP_{v,i}(S+T) = \sum_X AP^X_{v,i}(S+T) \cdot Prob[X]$ and similarly for $AP_{v,i}(T)$ and $AP_{v,i}(S|T)$. It remains to show that for any fixed X, we have

$$AP^X_{v,i}(S+T) - AP^X_{v,i}(T) \leq AP^X_{v,i}(S|T)$$

If $AP^X_{v,i}(T) = 1$, above inequality holds trivially. We just need to consider the case $AP^X_{v,i}(T) = 0$, or equivalently $d_X(T,v) \neq i$, where $d_X(T,v)$ denotes the distance between set T and node v under random edge insertion result X. If $d_X(T,v) < i$, then we have $d_X(S+T,v) \leq d_X(T,v) < i$ and thus $AP^X_{v,i}(S+T) - AP^X_{v,i}(T) = 0 \leq AP^X_{v,i}(S|T)$. Otherwise, if $d_X(T,v) > i$, we only need to consider the case when $AP^X_{v,i}(S+T) = 1$, namely $d_X(S+T,v) = i$. Since when $AP^X_{v,i}(S+T) = 0$, the inequality holds trivially. It holds that the shortest path from $S+T$ to v must not include any node in set T, otherwise it contradicts with the fact that $d_X(T,v) > i$. This fact implies that $AP^X_{v,i}(S|T) = AP^X_{v,i}(S+T) = 1$ following the definition of graph $G(T)$. □

Then we summarize the above inequality from iteration 0 to $|V-S|$ and derive a tighter upper for $\rho_S(T)$. To facilitate our presentation, we define $\mathbf{P}^G_{u,v}$ as the $|V|$-by-$|V|$ matrix of propagation probability $p^G_{u,v}$ associated with graph G. We denote $\boldsymbol{I}(S)$ as a $|V|$ dimension indicator vector, where $\boldsymbol{I}_v(S) = 1$ if and only if $v \in S$. We also organize $AP_{v,i}(\cdot)$ and $AP_{v,i}(\cdot|\cdot)$ into the $|V|$ dimension column vectors $\boldsymbol{AP}_i(\cdot)$ and $\boldsymbol{AP}_i(\cdot|\cdot)$.

Lemma 2. For any $v \in V$, and $S,T \subseteq V$, $S \cap T = \emptyset$, we have

$$\rho_S(T) \leq \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G(S+T)})^i \cdot \mathbf{1}$$

where T is for vector/matrix transpose and T is for the seed set.

Proof:

$$\begin{aligned}
\rho_S(T) &= \sigma(S+T) - \sigma(T) \\
&= \sum_{i=0}^{|V-S|} (\boldsymbol{AP}_i(S+T) - \boldsymbol{AP}_i(T)) \cdot \mathbf{1} \\
&\leq \sum_{i=0}^{|V-S|} \boldsymbol{AP}_i(S|T) \cdot \mathbf{1} \\
&\leq \boldsymbol{AP}_0(S|T)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G(S+T)})^i \cdot \mathbf{1} \\
&= \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G(S+T)})^i \cdot \mathbf{1}
\end{aligned}$$

□

It should be noticed that when taking $T = \emptyset$, we provide upper bound on $\sigma(S) = \rho_S(\emptyset)$ as a special case of Lemma 2, that is:

$$\sigma(S) \leq \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G(S)})^i \cdot \mathbf{1}$$

Compared to the upper bound derived by Zhou et al. in [13], namely

$$\sigma(S) \leq \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G})^i \cdot \mathbf{1}$$

we can easily see that our bound is tighter as $p^{G(S)}_{u,v} \leq p^G_{u,v}$. The reason is that the upper bound of $\sigma(S)$ in [13] actually computes all the possible influence paths within length of $|V-S|$ including loops and multiple pathes. In particular, they include the influence to nodes in S which they should not have as they are already activated at the beginning. On the contrary, we get a tighter upper bound by excluding the impossible influence paths including any edge (u,v) such that $v \in S$. To use our bounds in practice, we relax the summation to be taken on infinite steps as in done in [13]. As a result, we have

$$\begin{aligned}
\rho_S(T) &\leq \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{|V-S|} (\mathbf{P}^{G(S+T)})^i \cdot \mathbf{1} \\
&\leq \boldsymbol{I}(S)^{\mathrm{T}} \sum_{i=0}^{\infty} (\mathbf{P}^{G(S+T)})^i \cdot \mathbf{1} \\
&= \boldsymbol{I}(S)^{\mathrm{T}} \cdot (\mathbf{I} - \mathbf{P}^{G(S+T)})^{-1} \cdot \mathbf{1} \\
&= \boldsymbol{I}(S)^{\mathrm{T}} \cdot \boldsymbol{U}^{G(S+T)}.
\end{aligned}$$

where $\boldsymbol{U}^{G(S+T)} = (\mathbf{I} - \mathbf{P}^{G(S+T)})^{-1} \cdot \mathbf{1}$ is a column vector with length $|V|$. If we make the assumption that the condition $\lim_{i \to \infty} \mathbf{P}^{G(S+T)} = 0$ holds, the solution to the equation converges exponentially fast and can be computed iteratively as follows:

$$\boldsymbol{U}^{G(S+T)} = \mathbf{1} + \mathbf{P}^{G(S+T)} \cdot \mathbf{1} + (\mathbf{P}^{G(S+T)})^2 \cdot \mathbf{1} + \ldots$$

We run the iterations until the L_1-norm of $\boldsymbol{I}(S)^{\mathrm{T}} \cdot (\mathbf{P}^{G(S+T)})^k \cdot \mathbf{1}$ is within 10^{-3}. As mentioned in [13], ?In real-world social networks, the propagation probability is often very small. Thus, Condition (14) usually stands.? In our paper, the assumption we made will hold if the propagation probability is very small and when the i is close to infinity, there will be no node be activated. Similar to condition (14) in [13], $\lim_{i\to\infty}\mathbf{P}^{G(S+T)} = 0$ usually stands in real-world social networks so the upper bound can be easily computed as above formula and no expensive computations is needed.

4.2.2 Lower Bound of Marginal Gain

In this section, we illustrate the computation of the lower bound on the marginal gain $\rho_S(T)$. This part is not directly used in our algorithm, but it's used to improve our upper bound of the replacing gain as showed below.

Lemma 3. For $v \in V$, and $S, T \subseteq V$, we have the following inequation:

$$\sigma(S+T) - \sigma(T) \geq \sigma(S|C(T)) \tag{2}$$

where $C(T) = \{v | v \in V, d(T,v) < \infty\}$ is the set of nodes connected from the nodes in T and $\sigma(S|C(T))$ is the influence activated by the seed set S without propagating along any node in $C(T)$.

Proof: Considering the probability distribution of all possible influence propagations between each pair of nodes, we define $R^X(S)$ as the set of nodes that can be reached from nodes in S under the sample X. Hence, we have:

$$\sigma(S+T) - \sigma(T) = \sum_X \mathbb{P}(X) \cdot |R^X(S) - R^X(T)|$$

We denote $R^X(S|C(T))$ as the set of nodes that are activated under S without propagating along any node in $C(T)$ in the sample X. For $v \in R^X(S|C(T))$, we have $v \in R^X(S)$ and $v \notin C(T)$. And if $v \in R^X(T)$, then $v \in C(T)$ which is a contradiction. Clearly, $v \in R^X(S) - R^X(T)$ and it follows that:

$$\begin{aligned}\sigma(S+T) - \sigma(T) &\geq \sum_X \mathbb{P}(X) \cdot |R^X(S|C(T))| \\ &= \sigma(S|C(T))\end{aligned}$$

□

Lemma 4. For $v \in V$, and $S, T \subseteq V$, the lower bound of the marginal gain $\rho_S(T)$ is:

$$\rho_S(T) \geq \mathrm{I}(S)(\mathrm{E} + P(S + C(T))) \cdot \mathbf{1} \tag{3}$$

where $P(S+C(T))$ represents for the probability matrix for $G(S + C(T))$ and E for the identity matrix.

Proof: Using Lemma 3, we have:

$$\begin{aligned}\rho_S(T) &\geq \sigma(S|C(T)) \\ &= \sum_{i=0}^{|V-S|} \sum_{v\in V} AP_{v,i}(S|C(T)) \\ &\geq \sum_{v\in V} AP_{v,0}(S|C(T)) + \sum_{v\in V} AP_{v,1}(S|C(T)) \\ &= \mathrm{I}(S)(\mathrm{E} + P(C(T)) \cdot \mathbf{1}\end{aligned}$$

where we only consider the influence of one hop from the seed set S blocked by $C(T)$. Obviously, the inequality holds. □

Analogously, we define $L(S) = (\mathrm{E} + P(S)) \cdot \mathbf{1}$ as the lower bound column vector under seed set S. Let $p_{u,v}(S)$ be the probability of node u activating node u in $G(S)$ and we can calculate it as follow:

$$L_v(S) = 1 + \sum\nolimits_{(u,v)\in\mathcal{E}} p_{u,v}(S)$$

Readers may have noticed that lower bound is affected by the connectedness of the network and may ask that the lower bound derived above becomes meaningless when the underlying network is strongly connected so why should we still calculate the lower bound? Although when the underlying network is strongly connected, the lower bound of marginal gain is not that meaningful, however, as mentioned in [13], in real-world social networks, there are little strongly connected components so the lower bound is usually meaningful and leads to a tighter upper bound of node replacement gain.

Hence, based on the previous two section's result on the upper bound and the lower bound of the marginal gain, the replacing gain is bounded as follow:

$$\begin{aligned}\delta_{v,v_s} &= \sigma(S - v_s + v) - \sigma(S) \\ &= \rho_v(S - v_s) - \rho_{v_s}(S - v_s) \\ &\leq U_v(S - v_s + v) - \rho(S - v_s) \\ &\leq U_v(S - v_s + v) - L_{v_s}(C(S - v_s)))\end{aligned}$$

4.3 Fast Update of the Replacement Upper Bound

We have shown previously how to compute a tighter bound on the replacement gain for one static network with a fixed seed set S. However, as network changes constantly, we need to update the upper bound according to the changes in propagation probability. Moreover, as we include new node into the seed set S, we also need to update the upper bound as the propagation probability matrix $\mathbf{P}^{G(S+T)}$ also changes.

Let $\mathbf{\Delta P}$ be different in propagation probability between the two graphs G and G' associated with propagation probability matrices $\mathbf{P}$ and $\mathbf{P}'$, namely $\mathbf{\Delta P} = \mathbf{P} - \mathbf{P}'$.

The update of the bound boils down to the updating of column vector $\boldsymbol{U}^G$. Using the second order approximation of matrix inversion, we can update $\boldsymbol{U}^G$ approximately as follows:

$$\begin{aligned}\boldsymbol{U}^{G'} &= (\mathbf{I} - \mathbf{P}')^{-1} \cdot \mathbf{1} \\ &= (\mathbf{I} - \mathbf{P} - \mathbf{\Delta P})^{-1} \cdot \mathbf{1} \\ &\approx (\mathbf{I} - \mathbf{P})^{-1} \cdot \mathbf{1} + (\mathbf{\Delta P} + \mathbf{\Delta P} \cdot \mathbf{P} \\ &\quad + \mathbf{P} \cdot \mathbf{\Delta P} + \mathbf{\Delta P} \cdot \mathbf{\Delta P}) \cdot \mathbf{1} \\ &= \mathbf{U}^G + (\mathbf{\Delta P} + \mathbf{\Delta P} \cdot \mathbf{P} \\ &\quad + \mathbf{P} \cdot \mathbf{\Delta P} + \mathbf{\Delta P} \cdot \mathbf{\Delta P}) \cdot \mathbf{1}\end{aligned}$$

we can update the lower bound as follow

$$\begin{aligned} \boldsymbol{L}^{G'} &= (\mathbf{I}+\mathbf{P}')\cdot\mathbf{1} \\ &= (\mathbf{I}+\mathbf{P}+\mathbf{\Delta P})\cdot\mathbf{1} \\ &= (\mathbf{I}+\mathbf{P})\cdot\mathbf{1}+\mathbf{\Delta P}\cdot\mathbf{1} \\ &= \mathbf{L}^{G}+\mathbf{\Delta P}\cdot\mathbf{1} \end{aligned}$$

Our UBI algorithm only updates the upper bound and the UBI+ algorithm updates both the upper bound and the lower bound. Let $\Delta = \{(u,v)|\Delta P_{u,v} \neq 0\}$, and the updating algorithm for UBI and UBI+ is shown in Algorithm 4 and 5.

Algorithm 4 UpdateBoundForUBI($P, \Delta P, U$)

Initial $U' \leftarrow U$
for $(i,j) \in \Delta$ **do**
  // **For each node v in N that is not i**
  for $v \in N_{-}(i)$ **do**
    $U'_v += P_{v,i}\Delta P_{i,j}$
  end for
  $U'_i += \Delta P_{i,j} L_j$
end for
Output U'

As in UBI algorithm, we do not particularly calculate the lower bound so we just simply run Monte-Carlo simulations or use other heuristics to estimate L_j.

Algorithm 5 UpdateBoundForUBI+($P, \Delta P, U, L$)

Initial $U' \leftarrow U$
Initial $L' \leftarrow L$
for $(i,j) \in \Delta$ **do**
  // **For each node v in N that is not i**
  for $v \in N_{-}(i)$ **do**
    $U'_v += P_{v,i}\Delta P_{i,j}$
  end for
  $U'_i += \Delta P_{i,j} L_j$
  $L'_i += \Delta P_{i,j}$
end for
Output U' **and** L'

As long as $\mathbf{\Delta P}$ is sparse with only a few non-zero entries, the above formula leads to a fast update for the upper bound. It turns out that this is exactly the case in our setting. When we add a new node v to the seed set, we have at most d_v non-zero entries in $\mathbf{\Delta P}$, where d_v as the degree of node v is generally small and upper bounded by $|V|$. Moreover, when we change from one snapshot graph G^t to next time step G^{t+1}, the continuity of social network dynamics ensures that $\mathbf{\Delta P}$ is usually sparse. Additionally, for social network, matrix $\mathbf{P}$ is also sparse as usually the number of edges in social network is of the order $O(|V|)$. This fact further accelerates the computation of Equation (4).

Discussion As can be seen from the above algorithm, our UBI+ algorithm updates one more bound, the lower bound, than UBI. Because of the calculation of the lower bound, it may lead to a longer total running time. However, as shown later in experiments, UBI+ reaches a significant improvement of the influence spread. Compared with the influence improvement, the running time cost is relatively acceptable.

5 EXPERIMENTS

In this section, we conduct extensive experiments on three real-world dynamic large-scale networks to evaluate the performance of our algorithm for the INT problem.

5.1 Experiment Settings

First, we run our experiments on three real-world dynamic networks, Mobile, HepPh and HepTh to study UBI and UBI+'s performance on different scales. Results of these experiments is shown in Section 5.2.1 and 5.2.2. In Section 5.2.3, we run UBI and UBI+ on a benchmark for viral marketing to show our methods' performance on viral marketing.

Datasets. The first one, Mobile network used in [5] is extracted from mobile phone call records in a city during July 2007. Each node represents a mobile phone user and each phone call between two users creates a edge. The HepPh and HepTh provided in 2003 KDD cup are two citation networks extracted from the different sections of e-print arXiv[2]. In this network, each node corresponds to a paper instead of a author. We create an edge between node u and v, if paper v cites paper u. HepPh, HepTh and Mobile datasets are all directed networks.

The basic statistics of the three networks are summarized in Table 2. We use the following method to construct the snapshot graphs from the above datasets. At time stamp t, we generate the snapshot $G^t = (V, E^t), V = \bigcup V^t$ containing all the edges occurring in the time window $[t \cdot \Delta t, t \cdot \Delta t + \omega]$ where ω is the size of the observation window and Δt is the distance between two consecutive snapshots. Basically, ω controls the number of edges in each snapshot graph, while Δt decides the similarity between two consecutive snapshot graph. Thus by using different parameters ω and Δt, we can generate a family of snapshot graphs with different properties for our following experiments.

The number of edges in each snapshot graph generated from the networks is shown in Figure 2.

TABLE 2
Statistics of network datasets

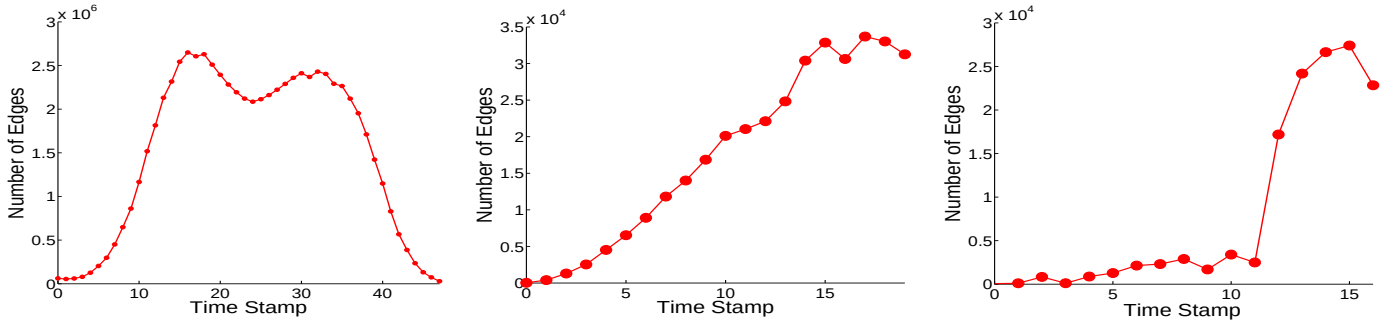
Dataset	Nodes	Edges	Date
Mobile	5.19M	12.0M	7 2007
HepPh	30.4K	346.9K	1992 - 2002
HepTh	18.5K	136K	1992 - 2002

Propagation probability. We assign the propagation probability on each edge by the following two widely-adopted models.

- **Uniform Activation (UA)**: UA model assigns probability uniformly. We set all the propagation probabilities to 0.05 in our experiments.
- **Degree Weighted Activation (DWA)**: DWA assigns probability of each edge (u,v) as $P_{u,v} = 1/d_{in}(v)$ where $d_{in}(v)$ is the in-degree of node v.

Algorithms under comparison. We compare UBI algorithm with the following state-of-the-art algorithms.

2. http://www.arXiv.arg



(a) Mobile (b) Hepph (c) Hepth

Fig. 2. Number of edges in snapshot graphs generated from three network datasets.

- **IMM**: IMM algorithm, which is a near-linear time greedy algorithm introduced in [20]. We run IMM algorithm for $\epsilon = 0.01$ as provided in the source code.
- **IRIE**: IRIE is the most advanced heuristic method under IC model. We run IRIE algorithm independently for each snapshot graph with parameters $\alpha = 0.7$ and $\theta = 1/320$ as reported in [17].
- **Degree**: As a baseline comparison, simply select the nodes with the highest degrees.
- **UBI**: Our UBI algorithm using SP1M [4] for influence estimation with $\gamma = 0.01$. The initial seed set S^0 is generated by **Greedy**. In UBI algorithm, we only calculate the upper bound of marginal gain when calculating the upper bound of node replacement gain.
- **UBI+**: Our UBI algorithm which calculates both the upper bound and the lower bound of the marginal gain when calculating the upper bound of node replacement gain.

We do not include other baseline methods for INT problem since it has already been shown that Greedy always has the best influence coverage while IRIE has slightly worse performance but runs significantly faster than other methods in time [17]. We use the average of 20000 rounds of Monte-Carlo simulations as estimation of the actual influence in order to evaluate the seed sets discovered by the algorithms. Moreover, all the experiments are carried out on a server with 32 cores (2.13G Hz) and 64G memory.

5.2 Experiment Results

5.2.1 Experiment Results of UBI

Influence coverage and running time on real dynamic networks. We first present our main result on comparing our UBI algorithm to other baseline methods on three real-world dynamic networks. For Mobile network, we set the window size to one hour while the time difference is set to two minutes. For both HepPh and HepTh network, we set the window size to three years and the time difference to one month. Moreover, we choose the seed size k as 30.

The results on influence coverage of the selected seed sets for each snapshot graph are shown in Figure 3 and Figure 4. As Greedy is too slow to finish within a reasonable time, we do not include Greedy on Mobile dataset.

TABLE 3
Average influence spread in UA Model

Dataset	Mobile	Hepph	Hepth
Greedy		71.49	65.49
IMM	95.42	71.24	65.43
IRIE	87.99	70.64	64.88
UBI	94.36	71.02	64.36

TABLE 4
Average influence spread in DWA model

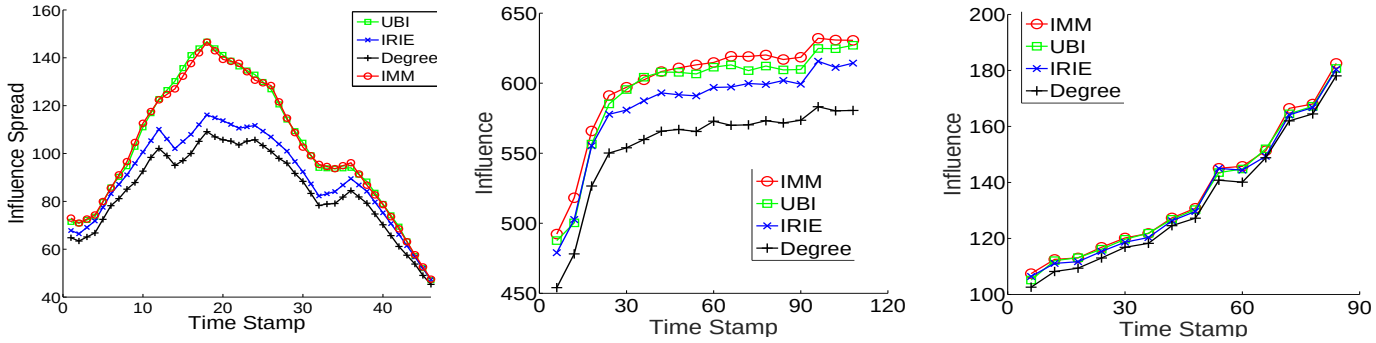
Dataset	Mobile	Hepph	Hepth
Greedy		124.35	74.81
IMM	1053.78	124.33	74.48
IRIE	943.69	122.94	74.32
UBI	1033.74	123.79	74.35

We also calculate the average influence spread over all snapshot graphs for all three networks and present the results in Table 3 and Table 4 for better comparison.

For the above results, we can easily find that UBI algorithm results in better influence coverage compared with IRIE averaged over all datasets. As our method has a little loss of accuracy on influence to achieve fast tracking, UBI achieves slightly lower influence compared to IMM and Greedy. Moreover, the running time taking average over different snapshot graphs for all three networks results of the above experiments are shown in Table 5 and Table 6.

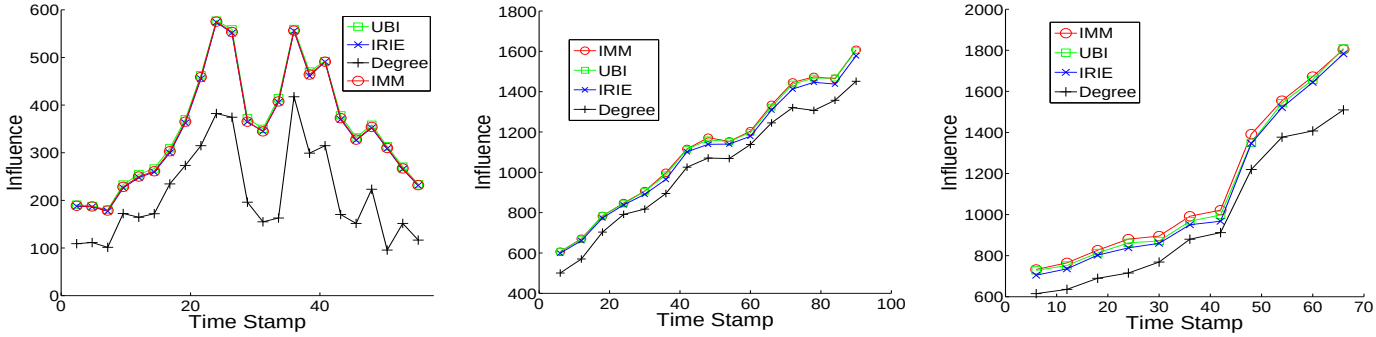
Reader may ask that if the influential users remain unchanged in most of real datasets, we do not have to track them with an online algorithm. To answer this question, we calculate the average influential users coverage of the result of UBI, which means the total number of users chosen to be the influential user of all time. Results are 151, 119, 143 for Mobile, Hepph and Hepth dataset. Because at every timestamp, the seed set only contains 30 nodes, the result reflects the fact that influential users vary frequently under the scenario of dynamic network.

We can easily find that Greedy is extremely slow that it even fails to finish on the largest Mobile network. Though performing well in influence coverage, IRIE performs well in running time on Hepth and Hepph but bad on Mobile with million nodes and edges. IMM performs better than IRIE on Mobile. However, our method, UBI, achieves consistent lowest running time on all the three networks with comparable influence coverage compared with IRIE,



(a) Mobile (b) HepPh (c) HepTh

Fig. 3. Influence Tracking Results under UA model with $k = 30$.



(a) Mobile (b) HepPh (c) HepTh

Fig. 4. Influence Tracking Results under DWA model with $k = 30$.

TABLE 5
Statistics of Running Time for UA Model

Dataset Running Time	Mobile	Hepph	Hepth
Greedy		12m	8m
IMM	1.8s	38ms	32ms
IRIE	19.1s	50ms	37ms
UBI	1.1s	22ms	17ms

TABLE 6
Statistics of Running Time for DWA Model

Dataset Running Time	Mobile	Hepph	Hepth
Greedy		53m	42m
IMM	2.1s	50ms	46ms
IRIE	30.3s	80ms	60ms
UBI	1.2s	21ms	16ms

IMM and Greedy algorithm. UBI is about 30 times faster than IRIE and 2 times faster than IMM. Notice that UBI achieves insignificant improvement compared with IRIE and IMM under the last two dataset. This is because they are in relatively small size. At the same time, as the size of networks grows, UBI can scale to networks like Mobile with million nodes and edges as shown in the following experiment.

Memory usage. We measure the memory usage of each algorithm to measure the space complexity and the result is shown in Table 7 and Table 8. As the dataset is also stored in memory, we measure the memory usage when only loading the dataset into the memory, which is marked as "None" in the table. As it can be seen from the result, our UBI algorithm uses a little more memory than Greedy and IRIE and less memory than IMM. UBI uses some additional space to calculate the upper bound and the lower bound to reach a much better influence coverage, but UBI uses only linear additional space so the space complexity is acceptable.

TABLE 7
Statistics of Memory Usage for UA Model

Dataset Memory Usage	Mobile	Hepph	Hepth
None	2376.2MB	27.4MB	24.1MB
Greedy		27.4MB	24.1MB
IMM	2564.7MB	28.9MB	26.0MB
IRIE	2496.5MB	27.8MB	24.5MB
UBI	2536.7MB	29.5MB	26.3MB

TABLE 8
Statistics of Memory Usage for DWA Model

Dataset Memory Usage	Mobile	Hepph	Hepth
None	2375.1MB	27.3MB	24.0MB
Greedy		27.3MB	24.0MB
IMM	2522.7MB	28.5MB	25.2MB
IRIE	2493.7MB	27.9MB	24.6MB
UBI	2537.9MB	29.2MB	26.4MB

Varying K. As the third experiment, we test the algorithm with a large $K = 50$. We have the same setting except K as the first experiment. The results on influence coverage of

the selected seed sets for each snapshot graph are shown in Figure 5. As Greedy is too slow to finish within a reasonable time, we do not include Greedy on this experiment. Note that the results under the UA model are similar, which are not included in this paper due to the limited space.

From Figure 5, we have the similar conclusion that UBI has very close influence coverage compared to IMM, which is already proved in [20] that has consistently close influence coverage as Greedy when K is varying. Our method performs consistently while K is different.

Scalability. As the fourth experiment, we test the scalability of our method on networks with different size. We construct a family of snapshot graphs from the Mobile dataset by varying the time window size ω from $2, 4, ..., 512$ minutes with a fixed time difference $\Delta t = 2$ minutes. The average numbers of edges in these graphs vary from 15K to 4M. We run algorithms to track $k = 30$ influential nodes under both the UA and the DWA Model for propagation probability. The running time under DWA model is shown in Figure 6, with normal scale in Figure 6(a) and log-log scale of the same figure in Figure 6(b). The results under UA model are similar and omitted. We don't plot the running time of Greedy for the measurement of Greedy on Mobile network is unaffordable.

As it is shown in Figure 6, our UBI algorithm is one magnitude faster than the IRIE and achieves about 2x speed up compared to the IMM algorithm. It clearly demonstrates the scalability of our algorithm for INT problem under large-scale dynamic networks.

Random seed vs. S^{t-1} As the fifth experiment, we test the influence of using random seed or S^{t-1} when updating the influence vector. We run the two algorithms to track $k = 30$ influential nodes under both the UA and the DWA Model for propagation probability. The average influence spread under DWA and UA model is shown in Figure 8, while the running time is shown in Figure 9(The figure is in log scale). From the result, it can be clearly seen that using random seed and S^{t-1} to can reach the same influence spread with enough interchange times, but using random seed is about 10 times slower than just using S^{t-1}.

As is shown in Figure 7, using random seed can reach the same influence spread as using the greedy algorithm when the interchange time is 30 or above, while using S^{t-1} only needs about 5-7 times. It clearly demonstrates the efficiency gap between using random seed and using S^{t-1}.

Similarity v.s. Updating time. The efficiency of our UBI algorithm comes from the fact that we utilize the similarity between two consecutive snapshot graphs. To quantitatively characterize the speedup, we conduct an experiment to explore how the similarity of the consecutive graphs correlates with the updating time for UBI. We use the Jaccard similarity to measure the similarity between two consecutive snapshot graphs G^t and G^{t+1}. Formally, we have:

$$\text{Jaccard}(G^t, G^{t+1}) = \frac{|E^t \cap E^{t+1}|}{|E^t + E^{t+1}|}$$

By varying the time difference Δt from $1, 2, 4, ..., 64$ minutes with a fixed one hour window, we construct a series of snapshot graphs with different Jaccard similarity from the Mobile dataset.

Figure 10 shows how the average updating time of our UBI algorithm is related to the average Jaccard similarity. In line with out intuition, the more similar two consecutive snapshots graphs are, the less time it takes by UBI algorithm to update the seed set. Moreover, even under extremely low Jaccard similarity, where the current snapshot differs greatly from the previous one, our UBI algorithm can still achieve low updating time by utilizing the upper bound on the node replacement gain.

Upper bounds comparison. As we discussed in section 3, our upper bound termed as active nodes' path excluded upper bound (AB), is theoretically tighter than the upper bound proposed in [13], which we call it the naive upper bound (NB). In order to validate our theory, we run empirical experiments to compare our bound AB with the naive upper bound. We first extract a series of snapshot graphs from Mobile datasets by setting both time window and time difference to one hour. We run equivalent number of iterations in computing both AB and NB on the same node set with size $k = 30$ where propagation probabilities are set according to DWA model. The seed set is selected by Greedy algorithm that maximizes the influence under each snapshot. As is shown in Figure 9, our bound is consistently tighter than the naive bound proposed in [13] as suggested by our theory. It should be noticed that the poor performance of NB under DWA model is due to the fact that sometimes NB fails to converge in Mobile network.

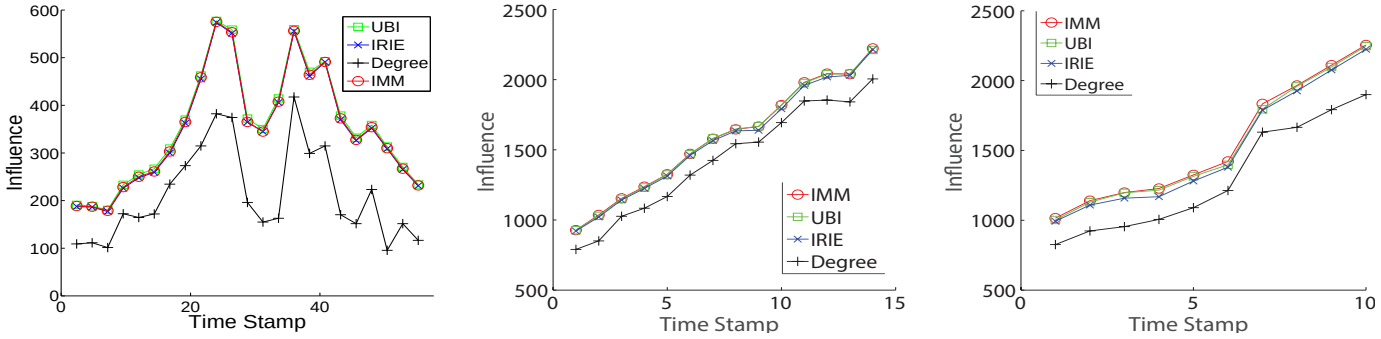
5.2.2 Experiment Results of UBI+

Influence coverage on dynamic networks. We present our result on comparing our improved UBI algorithm, UBI+ to UBI on three real-world dynamic networks. For Mobile network, we set the window size to one hour while the time difference is set to two minutes. For both HepPh and HepTh network, we set the window size to three years and the time difference to one month. Moreover, we choose the seed size k as 30. We calculate the average influence spread over all snapshot graphs for all three networks and present the results in Table 9 and Table 10. For the above results, we can easily find that our UBI+ algorithm achieves a better influence spread than UBI. Notice that UBI+ merely reaches about 2% and 1% better on the Hepph and Hepth dataset, this is because that UBI already performances very close to the influence spread upper bound(which is also the Greedy algorithm's result), so UBI+ only reaches an influence much closer to the theoretically influence bound. However, UBI+ get a 10% improvement in Mobile dataset and this shows that our new algorithm significantly improves the result in large datasets. Similar to the experiment results of UBI, the average influential users coverage of UBI+ is are 154, 119, 143 for Mobile, Hepph and Hepth dataset.

TABLE 9
Average influence spread in UA Model

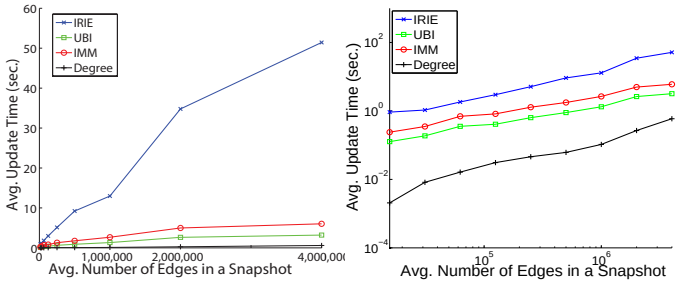
Dataset	Mobile	Hepph	Hepth
IMM	95.42	71.24	65.43
UBI	94.36	71.02	64.36
UBI+	95.01	71.15	65.04

Running time on dynamic networks. As it can be seen from Table 11 and Table 12, though being a little slower because



(a) Mobile (b) HepPh (c) HepTh

Fig. 5. Influence Tracking Results under DWA model with $k = 50$.



(a) Normal Scale (b) log-log Scale

Fig. 6. Scalability results on Mobile network with different number of edges in each snapshot graph.

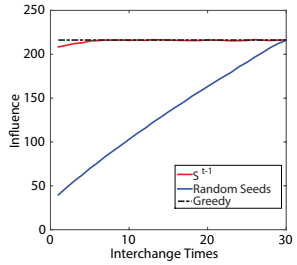
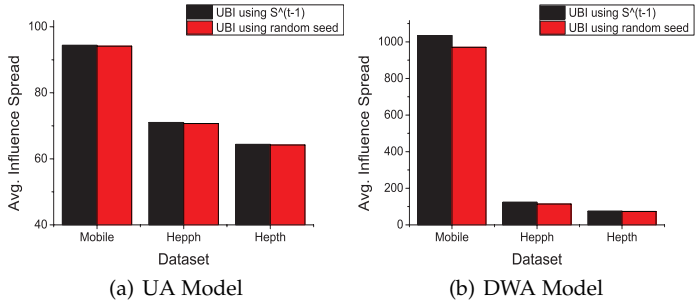
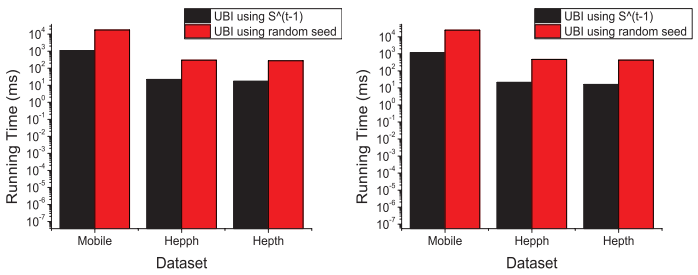


Fig. 7. An example illustrating UBI algorithm interchange from random seed set and S^{t-1}



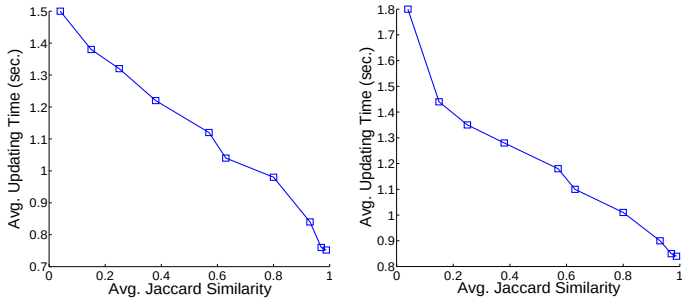
(a) UA Model (b) DWA Model

Fig. 8. Average influence spread of S^{t-1} vs. random seed



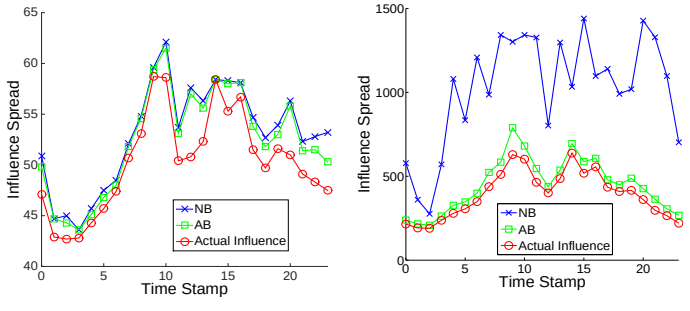
(a) UA Model (b) DWA Model

Fig. 9. Statistics of Running time of S^{t-1} vs. random seed



(a) UA Model (b) DWA Model

Fig. 10. Jaccard similarity vs. Updating time



(a) UA Model (b) DWA Model

Fig. 11. Actual influence spread compared with upper bounds

TABLE 10
Average influence spread in DWA model

Dataset	Mobile	Hepph	Hepth
IMM	1053.78	124.33	74.48
UBI	1033.74	123.79	74.35
UBI+	1045.15	124.01	74.42

that an additional bound need to be computed, UBI+ performs as well as UBI in running time. Notice that UBI+ achieves significant improvement in influence coverage in large datasets as the previous experiment shows, so a slight increase in the consumption of time is acceptable. So, in conclusion, UBI+ performs better than UBI in solving INT problem in large datasets. In small datasets, because that UBI already performs well so that UBI+'s improvement is not obvious.

TABLE 11
Statistics of Running Time for UA Model

Dataset Running Time	Mobile	Hepph	Hepth
IMM	1.8s	38ms	32ms
UBI	1.1s	22ms	17ms
UBI+	1.4s	25ms	21ms

TABLE 12
Statistics of Running Time for DWA Model

Dataset Running Time	Mobile	Hepph	Hepth
IMM	2.1s	50ms	46ms
UBI	1.1s	22ms	17ms
UBI+	1.5s	24ms	20ms

5.2.3 Experiment Results of UBI and UBI+ on viral marketing

Benchmark for viral marketing We use the benchmark proposed by Amit Goyal, etc. in [24] to measure our methods' performance. We generate a dataset by applying their benchmark algorithm to the Flixster dataset. The working principle of the benchmark is that the propagation probabilities between users in a social network can be learned from users' actions, such like making comments on movies, traveling to scenic spots, etc.. The Flixster dataset contains links between users and informations about which movie they've made comments on. The links in Flixster are undirected, but the influence probabilities learned is applicable for directed connections. As the result, the learned Flixter dataset with is a directed network, which contains 786.9k nodes and 4.7M edges.

Influence coverage We present our result on comparing our algorithms, UBI+ and UBI on the benchmark for viral marketing. We generate snapshot graphs from the flicker dataset generated by the benchmark mentioned in the previous section.

From Table 13, it can be seen that UBI and UBI+, similar to the results on HepPh, HepTh and mobile, achieves close influence spread to Greedy and IMM. This also supports our previous experiment results that UBI and UBI+ performs well in real dynamic networks.

TABLE 13
Average influence spread on benchmark for viral marketing(Flixster dataset)

Algorithm	Influence
Greedy	534.32
IMM	532.60
IRIE	524.14
UBI	529.16
UBI+	531.87

Running time. As it is shown in Table 14, though being a little slower because that an additional bound need to be computed, UBI+ performs as well as UBI in running time. Experiments result indicates that UBI and UBI+ runs much faster and proves our algorithm's ability to track influential users in a real, dynamic network. These experiments prove that our proposal works better on viral marketing.

TABLE 14
Statistics of Running Time on benchmark for viral marketing(Flixter dataset)

Algorithm	Running Time
Greedy	3h14m
IMM	1.23s
IRIE	13.15s
UBI	0.69s
UBI+	0.94s

6 CONCLUSIONS AND FUTURE WORK

In this paper, we explore a novel problem, namely Influential Node Tracking problem, as an extension of Influence Maximization problem to dynamic networks, which aims at tracking a set of influential nodes dynamically such that the influence spread is maximized at any moment. We propose an efficient algorithm UBI to solve the INT problem based idea of the Interchange Greedy method. We utilize the upper bound on node replacement gain to accelerate the process. Moreover, an efficient method for updating the upper bound is proposed to handle the evolution of the network structure. Extensive experiments on three real social networks show that our method outperforms state-of-the-art baselines in terms of both influence coverage and running time. Then we propose UBI+ algorithm that improves the computation of the upper bound and achieves better influence spread.

As a direct future work, we would like to generalize our UBI algorithm to track influential nodes under the other widely adopted diffusion model, Linear Threshold model under dynamic networks. Moreover, it will be interesting if we can combine our work with [21]. That is to track a series of influential nodes where the diffusion process is also carried out under a dynamic network instead of the static snapshot graph.

ACKNOWLEDGMENTS

This work was supported by the National High Technology Research and Development Program of China

(2014AA015103), the National Science and Technology Support Plan (2014BAG01B02), the National Natural Science Foundation of China (61572041), and the Beijing Natural Science Foundation (4152023).

REFERENCES

[1] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social network," in *KDD*, 2009, pp. 199–208.

[2] P. Domingos and M. Richardson, "Mining the network value of customers," in *KDD*, 2001, pp. 57–66.

[3] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *KDD*, 2003, pp. 137–146.

[4] M. Kimura and K. Saito, "Tractable models for information diffusion in social networks," in *PKDD*, 2006, pp. 259–271.

[5] W.Yu, G.Cong, G.Song, and K.Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *KDD*, 2010, pp. 1039–1048.

[6] W.Chen, C.Wang, and Y.Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *KDD*, 2010, pp. 1029–1038.

[7] W. Chen, W. Lu, and N. Zhang, "Time-critical influence maximization in social networks with time-delayed diffusion process," in *AAAI*, 2012.

[8] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 88–97.

[9] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha, "Scalable influence estimation in continuous-time diffusion networks," in *Advances in neural information processing systems*, 2013, pp. 3147–3155.

[10] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *KDD*, 2005, pp. 177–187.

[11] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *TKDD*, vol. 1, 2007.

[12] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins, "Microscopic evolution of social networks," in *KDD*, 2008, pp. 462–470.

[13] C. Zhou, P. Zhang, J. Guo, X. Zhu, and L. Guo, "Ublf: An upper bound based approach to discover influential nodes in social networks," in *ICDM*, 2013.

[14] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *KDD*, 2002, pp. 61–70.

[15] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. S. Glance, "Cost-effective outbreak detection in networks." in *KDD*, 2007, pp. 420–429.

[16] Q. Jiang, G. Song, G. Cong, Y. Wang, W. Si, and K. Xie, "Simulated annealing based influence maximization in social network," in *AAAI*, 2011.

[17] K. Jung, W. Heo, and W. Chen, "Irie: Scalable and robust influence maximization in social networks," in *ICDM*, 2012, pp. 918–923.

[18] M. G. Rodriguez and B. Schölkopf, "Influence maximization in continuous time diffusion networks," *arXiv preprint arXiv:1205.1682*, 2012.

[19] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 75–86.

[20] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: a martingale approach," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 2015, pp. 1539–1554.

[21] C. C. Aggarwal, S. Lin, and S. Y. Philip, "On influential node discovery in dynamic social networks." in *SDM*, 2012, pp. 636–647.

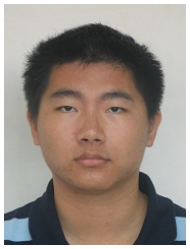
[22] H. Zhuang, Y. Sun, J. Tang, J. Zhang, and X. Sun, "Influence maximization in dynamic social networks," in *ICDM*, 2013, pp. 1313–1318.

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

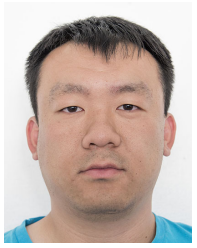
[24] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *Proceedings of the VLDB Endowment*, vol. 5, no. 1, pp. 73–84, 2011.

Guojie Song received the PhD degree from Peking University, Beijing, China, in 2004. He is currently an associate professor with the School of Electronic Engineering and Computing Science and the vice director in the Research Center of Intelligent Information Processing, Peking University. His research interests include various techniques of data mining, machine learning and their applications in intelligent transportation systems, and social networks.

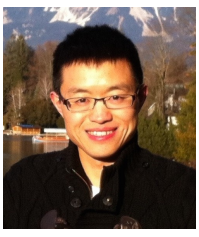
Yuanhao Li , undergraduate student in the Computer Science Department of Peking University. He is under advising of professor Prof. Guojie Song. His research interests lies in techniques of data mining, machine learning and their applications in social networks, and artificial intelligence.



Xiaodong Chen , Master student in the Computer Science Department of Peking University. He is in Key Laboratory of Machine Perception (Ministry of Education) of Peking University. His research interests include techniques of data mining, machine learning and their applications in intelligent transportation systems, and social networks.



Xinran He , PhD student in the Computer Science Department of University of Southern California. His research interest lies in social network analysis and social media analysis.



Jie Tang received the PhD degree from Tsinghua University, Beijing, China. He is an associate professor in the Department of Computer Science and Technology, Tsinghua University. His main research interests include data mining and social network analysis. He has been a visiting scholar at Cornell University, Chinese University of Hong Kong, Hong Kong University of Science and Technology, and Leuven University. He is a senior member of the IEEE.