# Time Constrained Influence Maximization in Social Networks

Bo Liu
Facebook
Menlo Park, CA, USA
bol@fb.com

Gao Cong, Dong Xu
School of Computer Engineering
Nanyang Technological University, Singapore
{gaocong, dongxu}@ntu.edu.sg

Yifeng Zeng
School of Computing
Teesside University, UK
Y.Zeng@tees.ac.uk

***Abstract*—Influence maximization is a fundamental research problem in social networks. Viral marketing, one of its applications, is to get a small number of users to adopt a product, which subsequently triggers a large cascade of further adoptions by utilizing "Word-of-Mouth" effect in social networks. Influence maximization problem has been extensively studied recently. However, none of the previous work considers the time constraint in the influence maximization problem.**

**In this paper, we propose the time constrained influence maximization problem. We show that the problem is *NP-hard*, and prove the monotonicity and submodularity of the time constrained influence spread function. Based on this, we develop a greedy algorithm with performance guarantees. To improve the algorithm scalability, we propose two Influence Spreading Path based methods. Extensive experiments conducted over four public available datasets demonstrate the efficiency and effectiveness of the Influence Spreading Path based methods.**

## I. Introduction

The influence maximization problem has been extensively studied (e.g., [1]–[8]). It aims to find a set of $K$ influential nodes such that the expected number of nodes reached by influence spreading from the selected node set is maximized. Among others, a motivating application of influence maximization is viral marketing in social networks (e.g., Facebook), which has become a common ground for businesses to target potential customers. Viral marketing aims to select a small number of influential users to adopt a product, and subsequently trigger a large cascade of further adoptions by utilizing the "Word-of-Mouth" effect in social networks [9], [10]. For example, a pop vocal concert marketer may select a small number of influential users of a social network, and offer each of them a free ticket, such that the concert is widely known throughout the entire social network.

Recent research reveals that time plays an important role in the influence spread from one user to another [11] and the time needed for a user to influence another varies. Indeed, influence propagation time is considered in the recent work [11]–[15] on building the underlying influence propagation graph from real world log data. On the other hand, in many real world viral marketing applications, people only care about how widely the influence is spread before a fixed time. For example, to market a pop vocal concert to be held on Sep 1st 2012, the marketer would want to maximize the number of users influenced before Sep 1st 2012. Indeed, users influenced after the concert would not bring any profit to the marketer.

This work was done while Bo Liu worked at Nanyang Technological University. Gao Cong is supported in part by a grant awarded by Microsoft Research Asia and by a Singapore MOE AcRF Tier 1 Grant (RG16/10). Dong Xu is supported by the National Research Foundation Singapore under its Interactive&Digital Media (IDM) Public Sector R&D Funding Initiative (Grant No. NRF2008IDM-IDM004-018) and administered by the IDM Programme Office.

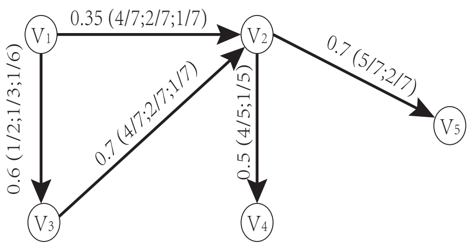


Fig. 1. An example illustrating the time constrained influence maximization.

However, none of the previous works considers the influence maximization under the time constraint. [1] We proceed to illustrate the idea of incorporating time factor in influence maximization using an example in Figure 1. In this example, five users are connected by five edges, each of which indicates a user may influence over another user. Numbers over each edge give the corresponding influencing probabilities, and the distribution of influencing delays. For example, user $v_2$ will influence $v_5$ with a probability of 0.7, and the influencing delay is distributed among the first two time units (e.g., day) with probability $5/7$ and $2/7$ respectively. This means that user $v_2$ would influence $v_5$ within the first time unit (resp. the second time unit) at a probability $0.7 \times 5/7$ (resp. $0.7 \times 2/7$), and $v_2$ cannot influence $v_5$ after the first two time units. Suppose we are asked to find a single seed user to maximize the expected number of influenced users. Without any time constraint, user $v_1$ will be returned as the result since it is expected to influence 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 in 1 time unit, user $v_2$ will be returned as the result (the algorithms for calculating the result will be presented in later sections).

[1] We note that the problem of time-constrained influence maximization is independently studied by Chen et al. [16].

In this paper, we define the time constrained influence maximization problem, which is based on the Latency Aware Independent Cascade influence propagation model. We prove that the time constrained influence maximization problem is *NP-hard*. We propose an algorithm that considers time factor in the process of Monte Carlo simulation to estimate the influence spread for a given seed set. This enables us to employ a greedy algorithm to solve the time-constrained influence maximization problem. However, the greedy algorithm is computationally expensive, which does not complete the task after two days running with LiveJournal dataset (to be described in Section V). As to be discussed in Related Work section, existing solutions for conventional influence maximization problem do not consider time factor. To this end, we propose two Influence Spreading Path based methods to solve the time constrained influence maximization problem.

The contributions of this paper are summarized as follows.

- We define the time constrained influence maximization problem in social networks. We prove the *NP-hardness* of the problem. We study the monotonicity and submodularity of the corresponding time constrained influence spread function. We also propose a time step based simulation algorithm for estimating the time constrained influence. These lead to a simulation based approximate algorithm.
- To develop more computation and memory efficient algorithms, we propose to logically augment a social network to incorporate influencing delay information, and define the Influence Spreading Path, based on which we propose two algorithms.
- Extensive experiments conducted on four public available datasets demonstrate the efficiency and effectiveness of the Influence Spreading Path based methods, and show that the influential nodes returned by methods of solving conventional influence maximization problem incur low influence for the time constrained version.

The remainder of this paper is organized as follows. The related work is reviewed in the next section. In Section III, we present a latency aware independent cascade model and the formal problem definition. In Section IV, we give a greedy algorithm, and then propose a simulation and two Influence Spreading Path based solutions. Section V presents the experimental study. Finally, Section VI concludes this paper and points out future directions.

## II. Related Work

### A. Influence Propagation Graph

The problem of building the underlying influence propagation graph has been studied recently. Saito et al. [14] propose an asynchronous model to extend the traditional Independent Cascade Models by incorporating influence spreading delay information. The proposed asynchronous model is employed to facilitate model parameter learning of the influence graph. Other efforts of learning parameters of the influence graph from history data include the work [11], [13]. In fact, the problem of building an influence graph is orthogonal to influence maximization problem, which assumes that the influence graph is known.

### B. Influence Maximization

Richardson et al. [1], [2] are the first to study influence maximization problem in social networks. They formulate the problem with a probabilistic framework and employ Markov Random Field to solve it. Kempe et al. [3] formulate the problem as a discrete optimization problem, which is widely adopted by subsequent studies. They prove the influence maximization problem is *NP-hard*, and propose a greedy algorithm to approximately solve it by repeatedly selecting the node incurring the largest marginal influence increase to a seed set. To find the node incurring the largest marginal influence increase at each step, one needs to know influence spreads induced by different seed sets generated by adding each individual candidate node into current seed set.

However, the problem of calculating influence spread induced by a given seed set is very difficult (Chen et al. [5] prove it to be *#P-hard*), and thus Kempe et al. [3] propose to simulate influence spreading process starting from the given seed set for a large number of times, and then use the average value of simulation results to approximate it. However, the simulation based method is computationally expensive and cannot scale well with large social networks [4]–[6]. To ease this problem, Leskovec et al. [4] propose a mechanism called Cost-Effective Lazy Forward (CELF) to reduce the number of times required to calculate influence spread, which will be used to optimize our algorithms in the conducted experiments. Chen et al. propose two fast heuristics algorithms, DegreeDiscount [5] and PMIA [6], to select nodes at each step of the greedy algorithm. At each step, DegreeDiscount adds the node with the largest degree to seed set, and then degrees of neighbors of the selected node are discounted accordingly. PMIA calculates influence spread by employing local influence arborescences, which are based on the most probable influence path between two nodes. As PMIA needs to maintain arborescence for each node, it consumes a huge amount of memory, which makes it unscalable to large social graphs. We compare with DegreeDiscount and PMIA in our experimental studies. Experimental results show that DegreeDiscount achieves much less influence spread compared to methods proposed in this work. PMIA achieves less influence spread than methods proposed in this work, and consumes much more memory, which makes it inapplicable to large social graph (LiveJournal dataset). In addition, Wang et al. [7] solve the problem by exploring the underlying community structure of social networks. Chen et al. [16] independently propose the time-critical influence maximization problem, in which the influencing model is a special case of the model proposed in this paper. In their model influence delays are constrained to follow the geometric distribution. In contrast, our model has no such a constraint and our algorithm is applicable when other distributions are used in the influencing model.

## III. Influence Propagation Model and the Time Constrained Influence Maximization Problem

We briefly describe the Independent Cascade (IC) model in Section III-A. We present the proposed Latency Aware Independent Cascade (LAIC) model, which incorporates influence propagation latency into the IC model in Section III-B. Based on the LAIC model, the formal definition of time constrained influence maximization problem is given in Section III-C. Notations used in this paper are summarized in Table I.

TABLE I
Notation Table

| Notation | Definition |
|---|---|
| $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ | Social Network |
| $n$ | $\lvert\mathcal{V}\rvert$ |
| $m$ | $\lvert\mathcal{E}\rvert$ |
| $K$ | Number of seed nodes |
| $S$ | Seed set |
| $N(u)$ | Neighbor set of $u$ |
| $\mathcal{P}_{uv}$ | Probability $u$ activates $v$ |
| $\mathcal{P}_u^{lat}$ | Distribution of influence propagation latency of $u$ |
| $\sigma_T(S)$ | Expected number of nodes influenced by $S$ within $T$ time units |
| $\mathcal{AP}^{(t)}(u,S)$ | Probability $u$ is first activated by $S$ at time $t$ |
| $\mathcal{AP}_T(u,S)$ | Probability $u$ is activated within time $T$ by $S$ |
| $ISP(S)$ | All influence spreading paths |
| $ISP(u,S)$ | All influence spreading paths ending with $u$ |
| $ISP_{\theta,T}(S)$ | Influence spreading paths with length no larger than $T$, probability no less than $\theta$ |
| $ISP_{\theta,T}(u,S)$ | Influence spreading paths with length no larger than $T$, probability no less than $\theta$, and ending with $u$ |
| $n_{\theta T}$ | $max_{\lvert S\rvert \le K}\{\lvert ISP_{\theta,T}(S)\rvert\}$ |

### A. Independent Cascade Model

Independent Cascade (IC) Model [18] is a popular model describing how influence spreads in social networks, which is widely adopted by the existing influence maximization algorithms [3]–[8].

In the IC model, each node of the social network is in either *active* (e.g., buying a product) or *inactive* state. A node is allowed to switch from *inactive* to *active* state, but not vice versa. Given a set of seed nodes $S$, the IC model propagates influence in inductive steps. Let $A_t$ be the set of nodes activated at step $t$, and $A_0 = S$. At step $t+1$, every node $u \in A_t$ has a single chance to activate each of its currently *inactive* neighbors $v$, i.e., $v \notin \cup_{i=0}^{t} A_i$. The probability that $u$ activates $v$ is given by the activating probability $\mathcal{P}_{uv}$ associated with edge $(u,v)$. The influence propagation process terminates at step $t$, if and only if $A_t = \phi$. In the IC model, once a node $u$ is activated, it either activates its currently *inactive* neighbor $v$ in the immediate next step, or does not activate $v$ at all.

However, as discussed in Section I, influence propagation delay exists in real world social networks, which is not captured by the IC model. We proceed to present the Latency Aware Independent Cascade (LAIC) model, which encodes the influence propagation latency information into the IC model.

### B. Latency Aware Independent Cascade (LAIC) Model

For LAIC model, when a node $u$ is first activated at step $t$, it activates its currently *inactive* neighbor $v$ in step $t + \delta_t$ with probability $\mathcal{P}_{uv}\mathcal{P}_u^{lat}(\delta_t)$, where $\delta_t$ is the influencing delay and is randomly drawn from the delay distribution $\mathcal{P}_u^{lat}$. Note that a node can be activated at most once. If a node has multiple neighbors influencing it, it is activated at the earliest activation time while the rest activations are ignored. The influence propagation process terminates at step $t$, iff there is no node activated after $t$.

### C. Problem Definition and its NP-hardness

Based on the proposed LAIC model, we next present the time constrained influence maximization problem.

**Time Constrained Influence Maximization Problem.** Given a social network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, time bound $T$, positive integer $K < |\mathcal{V}|$, activating probability $\mathcal{P}_{uv} \in (0,1]$ for each $(u,v) \in \mathcal{E}$, and latency distribution $\mathcal{P}_u^{lat}$ for each $u \in \mathcal{V}$, find a seed set $S \subset \mathcal{V}$ of $K$ nodes, such that the expected number of nodes influenced by $S$ within $T$ time, $\sigma_T(S)$, is maximized under the LAIC model.

Analogous to the conventional influence maximization problem, the time constrained version is *NP-hard*, which is shown in Theorem 1.

*Theorem 1:* The time constrained influence maximization problem is *NP-hard*.

*Proof:* As a traditional influence maximization problem (known to be *NP-complete*) is the corresponding time constrained problem with unlimited time bound, we argue that the traditional influence maximization problem is a special case of the time constrained influence maximization problem, which thus is *NP-complete*.

■

## IV. Influence Spreading Path based Solution

In Section IV-A, we show the *Monotonicity* and *Submodularity* of the time constrained influence function $\sigma_T(S)$ in the LAIC model, which thus leads to a natural hill climbing greedy algorithm framework. To implement the greedy algorithm, we need a method to calculate the expected influence spread for a given seed set, whose special case has been shown to be *#P-hard* [6]. To approximately solve this *#P-hard* problem, we propose a simulation based algorithm in Section IV-B, and define the Influence Spreading Path in Section IV-C. In Section IV-D, an Influence Spreading Path based algorithm is given. Finally, an algorithm employing faster marginal influence spread estimation is proposed in Section IV-E.

### A. Monotonicity, Submodularity and Greedy Algorithm

Let $\sigma_T(S)$ be the expected number of nodes influenced by $S$ within $T$ time units. By replacing $\sigma(S)$ with $\sigma_T(S)$, we adapt the greedy algorithm [3] to approximately solve the time constrained influence maiximization problem, which is given in Algorithm 1.

The greedy algorithm repeatedly adds the node incurring the largest marginal influence increase to seed set $S$, until $|S| =$

**Algorithm 1:** Greedy Algorithm Framework

```
Input: $\mathcal{G}$, $T$, $K$, $\mathcal{P}_{uv}$ and $\mathcal{P}_u^{lat}$
Output: $S$
1 initialize $S = \emptyset$
2 for $i \leftarrow 1$ to $K$ do
3     $u \leftarrow \arg\max_v \sigma_T(S \cup \{v\}) - \sigma_T(S)$
4     $S \leftarrow S \cup \{u\}$
5 end
6 return $S$
```

$K$. The time complexity of Algorithm 1 is $O(Kn\mathcal{T}(\sigma_T(S)))$, where $n$ is the number of nodes in $\mathcal{G}$ and $\mathcal{T}(\sigma_T(S))$ the running time for calculating $\sigma_T(S \cup \{v\})$. As Theorem 2 shows the influence function $\sigma_T(S)$ is monotonous and submodular, and thus the greedy algorithm approximates the optimal solution with a lower bound ratio of $1 - 1/e$ [20].

*Theorem 2:* With the LAIC model, the influence function $\sigma_T(S)$ is monotonous and submodular.

*Proof:* With the LAIC model, each social network can be treated as a random graph. Each edge $(u, v) \in \mathcal{E}$ is associated with a random *Bernoulli* variable governed by $\mathcal{P}_{uv}$, which controls the likelihood $u$ activates $v$. Each $u \in \mathcal{V}$ is associated with an influence delay distribution $\mathcal{P}_u^{lat}$, which controls the length of edges starting from $u$. Let $\mathcal{X}$ denote the entire probability space constituting all possible determined influence propagation graphs. A determined influence propagation graph is generated by flipping a coin of bias $\mathcal{P}_{uv}$ for every edge $(u, v) \in \mathcal{E}$ to determine if edge $(u, v)$ exists in the determined graph, and drawing a delay length from distribution $\mathcal{P}_u^{lat}$ for every node $u \in \mathcal{V}$ to determine the length of $u$'s outgoing edges in the determined graph. Then we have $\sigma_T(S) = \sum_{x \in \mathcal{X}} P(x)\sigma_{T,x}(S)$, where $P(x)$ is the probability of $x$, $\sigma_{T,x}(S)$ is the number of influenced nodes by $S$ within $T$ time units over the determined graph $x$.

For any determined graph $x \in \mathcal{X}$, $\sigma_{T,x}(S)$ is equal to the number of nodes reachable from $S$ by at least one path with length no larger than $T$. It is easy to find that $\sigma_{T,x}(S) \leq \sigma_{T,x}(S \cup \{u\})$, therefore $\sigma_{T,x}(S)$ is monotonous. As $\sigma_T(S) = \sum_{x \in \mathcal{X}} P(x)\sigma_{T,x}(S)$, and $P(x) \in (0, 1]$, $\sigma_T(S)$ is monotonous either.

Let $S_1 \subseteq S_2 \subseteq V$ and $u \in V$. We first consider a determined graph $x \in \mathcal{X}$. $\sigma_{T,x}(S_1 \cup \{u\}) - \sigma_{T,x}(S_1)$ is the number of nodes of $T$ length reachable from $u$, but not $T$ length reachable from $S_1$. As $S_1 \subseteq S_2$, we have $\sigma_{T,x}(S_1 \cup \{u\}) - \sigma_{T,x}(S_1) \geq \sigma_{T,x}(S_2 \cup \{u\}) - \sigma_{T,x}(S_2)$. Thus $\sigma_{T,x}(S)$ is submodular. Noticing that $\sigma_T(S)$ is a non-negtive linear combination of submodular functions $\sigma_{T,x}(S)$, Thus $\sigma_T(S)$ is submodular, which concludes the proof. ■

The main difficulty in applying the greedy algorithm lies in calculating the expected influence spread for a given set of seeds (Line 3 of Algorithm 1), whose special case has been shown to be *#P-hard* [6]. In the following sections, we propose a set of approximate algorithms including a simulation based algorithm and two Influence Spreading Path based algorithms.

## B. Simulation based Algorithm for $\sigma_T(S)$

We propose Algorithm 2 to simulate the time constrained influence spreading process based on time steps. Note that Algorithm 2 is different from the simulation algorithm for conventional influence maximization problem [3], which is based on Breadth-first Search (BFS) and does not consider time factor.

**Algorithm 2:** $\sigma_T(S)$ based on Simulation

```
Input: $\mathcal{G}$, $T$, $S$, $\mathcal{P}_{uv}$ and $\mathcal{P}_u^{lat}$
Output: $\sigma_T(S)$
1  $v.status \leftarrow$ inactive, $v.actTime \leftarrow +\infty$ for $v \in \mathcal{V} \setminus S$
2  $v.status \leftarrow$ active, $v.actTime \leftarrow 0$ for $v \in S$
3  $A_0 \leftarrow S$
4  $t \leftarrow 1$
5  do
6      for $u \in A_{t-1}$ do
7          for $(u, v) \in \mathcal{E}$ and $v.status \neq$ active do
8              draw $flag$ from $Bernoulli(\mathcal{P}_{uv})$
9              if $flag = 1$ then
10                 draw $\delta_t$ from $\mathcal{P}_u^{lat}$
11                 if $v.status =$ inactive then
12                     if $t + \delta_t \leq T$ then
13                         $v.status \leftarrow$ latent active
14                         $v.actTime \leftarrow t + \delta_t$
15                     end
16                 end
17                 else if $t + \delta_t < v.actTime$ then
18                     $v.actTime \leftarrow t + \delta_t$
19                 end
20             end
21         end
22     end
23     $A_t \leftarrow \{u | u.actTime = t \cap u.status =$ latent active$\}$
24     $u.status \leftarrow$ active for $u \in A_{t-1}$
25     $t \leftarrow t + 1$
26 while $|\{u | u.status =$ latent active$\}| \neq 0$ or $A_t \neq \emptyset$;
27 return $\sum_{j=0}^{t} |A_j|$
```

In Algorithm 2, we simulate the influence propagation process starting from $S$. In the beginning, all nodes in $S$ are set to be *active*, while all other nodes are set to be *inactive* (Lines 1-2 of Algorithm 2). The set of nodes activated at time $t$ is denoted by $A_t$. Nodes in $S$ are treated as being activated at time 0 (Line 3). At time step $t > 0$, each node $u \in A_{t-1}$ intends to activate each of its *inactive* or *latent active* (to be explained) outgoing neighbors $v \in N_{out}(u)$ with the probability $P_{uv}$. If $u$ successfully activates $v$ (Lines 9-20), an activating latency $\delta_t$ ($\delta_t = 0, 1, 2...$) is drawn from the discrete distribution $\mathcal{P}_u^{lat}$ associated with node $u$. If $v$ is in *inactive* state and $t + \delta_t \leq T$, $v$ switches to *latent active* state with activating time $t + \delta_t$, which specifies when $v$ will switch from *latent active* to *active*. If $v$ is already in *latent active* state, $v$ updates its activating time with the minimum of $t+\delta_t$ and its current activating time. All *latent active* nodes with activating time $t$ automatically switch to *active* state at time step $t$ (Lines 23-24). The process terminates if and only if there is no more *latent active* nodes and newly activated nodes. When the process terminates, the number of activated nodes is returned (Line 27).

**Time and space complexities** Let *n* (resp. *m*) be the number of nodes (resp. edges) in social network $\mathcal{G}$. The first four lines of Algorithm 2 take $O(n)$ time. For the entire *while* loop, the dominant cost is on exploring the graph starting from $S$ along edges. In the worst case, the algorithm needs to explore all nodes and edges in the graph. Thus the running time for the *while* loop is $O(n+m)$, which is also the time complexity of Algorithm 2. In addition to the input social graph, Algorithm 2 only needs to store $status$ and $actTime$ 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 social network.

To obtain an approximate value of the expected influence spread within $T$ time units, we need to repeat Algorithm 2 for a large number ($R$) of times and average the returned numbers. Consequently the total running time of the combination of Algorithm 1 and 2 is $O(KnR(n+m))$. By following [3], [5], [6], $R = 20,000$ simulations are employed to calculate the expected influence spread for a given seed set.

### *C. Influence Spreading Path based Activation Probability Calculation*

Due to the computational curse, the simulation based algorithm is not suitable to large social networks. We proceed to describe how a social network is augmented by incorporating influence delay information into the graph structure, based on which the definition of influence spreading path is given. Then we propose an algorithm for calculating the activation probability of a node given a seed set.

*1) Augmenting Social Network with Influencing Delay Information:* In the LAIC model, when a node $u$ is first activated at time $t$, it tries to activate each of its outgoing neighbors $v$ at a later time $t+\delta_t$ with a probability of $\mathcal{P}_{uv}\mathcal{P}_u^{lat}(\delta_t)$. To incorporate influence propagation delay information into the social network structure, we *logically* augment the original social network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ into a directed multigraph $G_T = (V, E)$, where $\mathcal{V} = V$. For each $(u, v) \in \mathcal{E}$, we put $T$ edges $e_{uv}^1, e_{uv}^2, \ldots, e_{uv}^T$ from $u$ to $v$ in $G$. Each edge $e_{uv}^t$ in $E$ is assigned with two values, i.e., $length(e_{uv}^t) = t$ and $prob(e_{uv}^t) = \mathcal{P}_{uv}\mathcal{P}_u^{lat}(t)$.

Figure 2 gives the multigraph augmented from the example of social network in Figure 1 under the case of $T = 2$. We note that this augmentation is done logically. All algorithms proposed in this paper are able to infer the augmented graph from an original graph on the fly.

*2) Constrained Influence Spreading Path:* Given a seed set $S$, the expected influence spread within time $T$, $\sigma_T(S)$, is the expected number of nodes activated no later than time $T$, denoted by $\sum_{u\in V}\mathcal{AP}_T(u, S)$, where $\mathcal{AP}_T(u, S)$ is the probability that $S$ activates $u$ within $T$. It is easy to find out

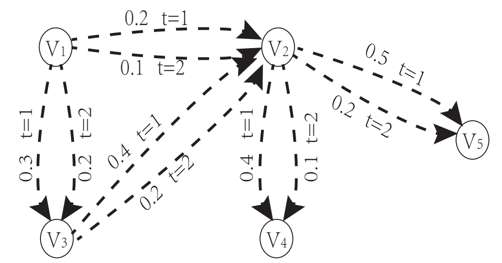


Fig. 2. The *logically* augmented multigraph with $T = 2$.

that $\mathcal{AP}_T(u, S) = 0$, if there is no path from $S$ to $u$ in the augmented directed multigraph $G_T = (V, E)$. Thus in what follows, we ignore those nodes not reachable from $S$.

To estimate $\mathcal{AP}_T(u, S)$ for each node $u$, we define Influence Spreading Path in the augmented graph below.

*Definition 1:* **Influence Spreading Path.** Given a seed set $S$ and a directed multigraph $G = (V, E)$, a simple path $p = (u_1 \xrightarrow{e_1} u_2 \xrightarrow{e_2} u_3 \ldots \xrightarrow{e_{k-1}} u_k)$ in graph $G$ is an Influence Spreading Path, if and only if $u_1 \in S$ and $u_i \notin S$ for $i \neq 1$, where $k > 1$. For an influence spreading path $p$, the length of $p$ is $\sum_{i=1}^{i=k-1} length(e_i)$, while the probability of $p$ is $\prod_{i=1}^{i=k-1} prob(e_i)$. Note that the algorithms proposed in this paper do **not** need the detailed path information, and we only need to store length, probability and the ending node of each Influence Spreading Path.

From Definition 1, we notice that an Influence Spreading Path cannot contain duplicate nodes, as a node cannot be activated more than once. Furthermore, except the starting point, an influence spreading path cannot contain any other nodes belonging to $S$, which comes from the fact that seed nodes are already in *active* state at the very beginning and cannot be activated at a later time.

We observe that each Influence Spreading Path $p$ ending with $u$ gives a possible way for $S$ to activate $u$. The activating time taken by following $p$ to activate $u$ is $length(p)$, while the activating probability of this path is $prob(p)$. For a given seed set $S$, we denote $ISP(u, S)$ to be all possible influence spreading paths ending with $u$. Note that $|ISP(u, S)|$ grows exponentially as the number of nodes increases. To reduce the number of paths in $ISP(u, S)$, we apply two restrictions to filter out some Influence Spreading Paths which are not or less related to our problem. First, we prune paths with length larger than $T$, which are not related to influence spread within time $T$. Furthermore, we filter out paths with probability less than a small threshhold $\theta > 0$, as Influence Spreading Paths with small probabilities have limited impacts on the influence spread estimation. The resulting constrained Influence Spreading Paths are denoted by $ISP_{\theta,T}(u, S)$.

*3) Activation Probability Calculation based on Influence Spreading Paths:* By assuming all Influence Spreading Paths ending at $u$ ($ISP_{\theta,T}(u, S)$) are independent with each other, we are able to calculate the probability $u$ gets activated by $S$ within time $T$ ($\mathcal{AP}_T(u, S)$) from $ISP_{\theta,T}(u, S)$ in Procedure $AP$ outlined below. Procedure $AP$ iterates over all possible time steps from 1 to $T$, calculates the probability that $u$ is first activated at time $t$ ($\mathcal{AP}^{(t)}(u,S)$) (Line 3), and adds it to $\mathcal{AP}_T(u,S)$ at Line 4. At Line 3, $1-\mathcal{AP}_T(u,S)$ is the probability $u$ has not been activated before $t$, and $1-\prod_{p\in ISP_{\theta,T}(u,S),length(p)=t}(1-prob(p))$ is the probability $u$ is activated at time $t$. At the end of each iteration $t$, $\mathcal{AP}_T(u,S)$ is updated to store the probability $u$ is activated before $t+1$. The loop results in the probability $\mathcal{AP}_T(u,S)$ that $u$ is activated within time $T$.

**Time and space complexities** For the running time, the dominant part of Procedure $AP$ is the *for* loop in which every Influence Spreading Path in $ISP_{\theta,T}(u,S)$ is checked exactly once. Thus the running time of Procedure $AP$ is $O(|ISP_{\theta,T}(u,S)|)$. Note that the space complexity of Procedure $AP$ is also $O(|ISP_{\theta,T}(u,S)|)$.

```
Procedure AP
  Input: $ISP_{\theta,T}(u,S)$, $T$
  Output: $\mathcal{AP}_T(u,S)$
1 $\mathcal{AP}_T(u,S) \leftarrow 0$
2 for $t \leftarrow 1$ to $T$ do
3     $\mathcal{AP}^{(t)}(u,S) \leftarrow (1-\mathcal{AP}_T(u,S))(1-\prod_{p\in ISP_{\theta,T}(u,S),length(p)=t}(1-prob(p))$
4     $\mathcal{AP}_T(u,S) \leftarrow \mathcal{AP}_T(u,S) + \mathcal{AP}^{(t)}(u,S)$
5 end
6 return $\mathcal{AP}_T(u,S)$
```

### *D. Influence Spreading Path based Algorithm for $\sigma_T(S)$*

Algorithm 3 computes the expected influence spread within time $T$ for a given seed set ($\sigma_T(S)$). First, Algorithm 3 gets all constrained Influence Spreading Paths starting from $S$ by a Depth-First Search (DFS) (Line 2), which are then divided into disjoint sets based on their ending nodes (Line 3). For each node $u$ with at least one constrained Influence Spreading Path, i.e., $ISP_{\theta,T}(u,S) \neq \emptyset$, Procedure $AP$ is applied to calculate the probability $\mathcal{AP}_T(u,S)$ that $u$ is activated by $S$ within time $T$ (Line 5). Finally, activation probabilities of all nodes are summed together and returned as the expected influence spread of $S$.

Like Algorithm 2, Algorithm 3 is embedded in Algorithm 1 (calculating $\sigma_T(S)$) to find a seed set of $K$ nodes.

**Time and space complexities** Let $n_{\theta T} = max_{|S|\leq K}\{|ISP_{\theta,T}(S)|\}$, where $|ISP_{\theta,T}(S)|$ is the number of Influence Spreading Paths starting from $S$ with length no less than $T$ and probability no less than $\theta$. The second line of Algorithm 3 can be done using DFS algorithm in $O(n_{\theta T})$ time, which is also the time needed for the third line. As calculating $\mathcal{AP}_T(u,S)$ by Procedure $AP$ takes $O(|ISP_{\theta,T}(u,S)|)$ time and $\sum_{u\in\mathcal{V}}|ISP_{\theta,T}(u,S)| = |ISP_{\theta,T}(S)| \leq n_{\theta T}$, the *for* loop also takes $O(n_{\theta T})$ time. Thus the total running time of Algorithm 3 is $O(n_{\theta T})$. Note that the Influence Spreading Path based solution (combination of Algorithm 1 and 3) takes $O(Knn_{\theta T})$ time, which is much less than the time needed by simulation based solution (combination of Algorithm 1 and 2) $O(KnR(n+m))$. It is obvious to see that the space complexity of Algorithm 3 is $O(n+m+n_{\theta T})$, where the $n+m$ part comes from the input social graph, and $n_{\theta T}$ is for storing $ISP_{\theta,T}(S)$.

```
Algorithm 3: $\sigma_T(S)$ based on Influence Spreading Path
  Input: $\mathcal{G}$, $\theta$, $T$, $S$
  Output: $\sigma_T(S)$
1 $\sigma_T(S) \leftarrow 0$
2 get all Influence Spreading Paths with length no larger than $T$ and probability no less than $\theta$ by DFS.
3 divide them into different $ISP_{\theta,T}(u,S)$.
4 for every $u$ with non-empty $ISP_{\theta,T}(u,S)$ do
5     $\sigma_T(S) \leftarrow \sigma_T(S) + AP(ISP_{\theta,T}(u,S),T)$
6 end
7 return $\sigma_T(S)$
```

### *E. Faster Marginal Influence Spread Estimation*

In the greedy Algorithm 1, when trying to add one more node into the currently selected seed set $S$, we need to calculate the marginal influence increase brought by adding each $u \in V\backslash S$. Instead of calculating $\sigma_T(S\cup\{u\})$ from scratch by Algorithm 3, we propose to employ faster marginal influence spread estimation.

Suppose currently selected seed set is $S$, we want to calculate the marginal influence spread increase if node $v$ is added to $S$, i.e., $\sigma_T(S\cup\{v\}) - \sigma_T(S)$, which is obviously no larger than $\sigma_T(\{v\})$. By noticing that $\sigma_T(\{v\})$ is already known as it was calculated when selecting the first seed node, we propose to approximate $\sigma_T(S\cup\{v\}) - \sigma_T(S)$ by making a discount of $\sigma_T(\{v\})$ in Equation 1.

$$\sigma_T(S\cup\{v\})-\sigma_T(S) \approx \sigma_T(\{v\})\frac{\sum_{(v,w)\in\mathcal{E}}\mathcal{P}_{vw}(1-\mathcal{P}_{Sw})\sigma_T(\{w\})}{\sum_{(v,w)\in\mathcal{E}}\mathcal{P}_{vw}\sigma_T(\{w\})} \tag{1}$$

where $\mathcal{P}_{Sw} = 1-\prod_{(u,w)\in\mathcal{E},u\in S}(1-\mathcal{P}_{uw})$ if $w \in N(S)$; otherwise, $\mathcal{P}_{Sw} = 0$. In other words, $\mathcal{P}_{Sw}$ is the probability $w$ gets immediately activated by seed nodes. The rationality behind Equation 1 is that the marginal influence increase is a discount of $\sigma_T(\{v\})$. The higher probability $v$'s neighbors are already activated by $S$, the larger discount should be applied to $\sigma_T(\{v\})$. With this marginal influence spread increase approximation, we propose Algorithm 4 to solve the time constrained influence maximization problem.

Algorithm 4 calculates time constrained influence spread based on influence spreading paths for each single node (Line 1-3). Seed nodes are selected by picking the node with the largest discounted marginal influence one by one (Lines 8-12).

**Time and space complexities** As Algorithm 3 takes $O(n_{\theta T})$ time, the first *for* loop of Algorithm 4 takes $O(nn_{\theta T})$ time. Line 9 takes $O(ne_{max})$ time while line 11 takes $O(Ke_{max})$ time, where $e_{max}$ is the largest degree among all nodes. Thus

**Algorithm 4:** Marginal Discount of Influence Spread Path

```
Input: $\mathcal{G}$, $T$, $K$, $\mathcal{P}_{uv}$, $\mathcal{P}_u^{lat}$, $\theta$
Output: $S$
1  for every $u \in V$ do
2      calculate $\sigma_T(\{u\})$ by Algorithm 3.
3  end
4  $u \leftarrow \arg\max_v \sigma_T(\{v\})$
5  $S \leftarrow \{u\}$
6  $\mathcal{P}_{Sw} \leftarrow \mathcal{P}_{uw}$ for $w \in N_{out}(u)$
7  $\mathcal{P}_{Sw} \leftarrow 0$ for $w \notin N_{out}(u)$
8  for $k \leftarrow 1$ to $K-1$ do
9      $u \leftarrow \arg\max_v \sigma_T(\{v\}) \frac{\sum_{(v,w)\in\mathcal{E}} \mathcal{P}_{vw}(1-\mathcal{P}_{Sw})\sigma_T(\{w\})}{\sum_{(v,w)\in\mathcal{E}} \mathcal{P}_{vw}\sigma_T(\{w\})}$
10     $S \leftarrow S \cup \{u\}$
11     update $\mathcal{P}_{Sw}$ for every $w \in N_{out}(S)$.
12 end
13 return $S$
```

the second *for* loop takes $O((K-1)ne_{max})$ time, and the total running time of Algorithm 4 is $O(n(n_{\theta T} + (K-1)e_{max}))$. Note that Algorithm 4 itself solves the time constrained influence maximization problem, and does not need to be combined with Algorithm 1. By comparing the time complexities of Algorithm 4 and the combination of Algorithms 1 and 3, whose running time is $O(Knn_{\theta T})$, we find that they have the same running time when $K = 1$, and Algorithm 4 runs faster when $K > 1$. This observation is consistent with the experimental results that will be presented in the next section. The memory space needed by Algorithm 4 is dominated by running Algorithm 3 at line 2, and thus the space complexity for Algorithm 4 is the same as that for Algorithm 3, which is $O(n + m + n_{\theta T})$.

## V. Experiments

### A. Experimental Setup

*1) Experimental Datasets:* Four public available[2] real-world social networks are used in the experiments. The basic statistics of these networks are summarized in Table II. The first one is a Wikipedia voting network, denoted by Wiki, where nodes represent wikipedia users, and an edge from node $i$ to $j$ represents that user $i$ voted on user $j$. The network contains all Wikipedia voting data from the inception of Wikipedia till Jan 3 2008. The second one is a who-trust-whom social network of a general consumer review site Epinions.com, which is denoted by Epinions. The third one is denoted by Slashdot, which is a social network extracted from the user community of Slashdot.org in February 2009. The last one is a large social network formed by LiveJournal community, denoted by LiveJournal.

*2) Evaluated Methods:* We note that all methods proposed in this paper are based on the greedy algorithm framework. The difference lies in the way of calculating the marginal influence increase, i.e., Line 3 of Algorithm 1. The following methods are evaluated.

- Monte Carlo (MC). Calculate both $\sigma_T(S \cup \{v\})$ and $\sigma_T(S)$ by simulations (combination of Algorithm 1 and 2). $20,000$ simulations are employed for each seed set by following [3], [5], [6].
- Influence Spreading Path (ISP). Calculate both $\sigma_T(S \cup \{v\})$ and $\sigma_T(S)$ by using Influence Spreading Paths (combination of Algorithm 1 and 3). The Influence Spreading Paths starting from each seed set are calculated from scratch by DFS.
- Marginal Discount of Influence Spread Path (MISP). Calculate influence spread $\sigma_T(u)$ for each single node $u$ with Influence Spreading Paths starting from $u$, then select seed node with the largest discounted marginal influence spread one by one (Algorithm 4).
- Random. Randomly select $K$ nodes as seed, which acts as the baseline method.
- Degree Discount (DC). The degree discount heuristic proposed by [5]. The implementation of DC used in this paper is provided by its authors.
- Prefix excluding Maximum Influence Arborescence (PMIA). PMIA [6] is a state-of-the-art solution for conventional influence maximization problem. The implementation of PMIA used in this paper is provided by its authors.

Note that all evaluated methods are enhanced by CELF [4] optimization if applicable.

*3) Parameter Setting:* The activating probability $\mathcal{P}_{uv}$ of each edge $(u, v)$ is set by the "Weighted Cascade" policy, which is widely adopted by the existing conventional influence maximization techniques [3], [5], [6]. With "Weighted Cascade" policy, $\mathcal{P}_{uv}$ is set to be $\frac{1}{N_{in}(v)}$, where $N_{in}(v)$ is the indegree of $v$.

The influencing delays ($\mathcal{P}_u^{lat}$) used in the experiments follow the Poisson distribution. For each node $u \in \mathcal{V}$, the parameter for its Poisson distribution (expected number of occurrences in a given interval) is randomly selected from the set $\{1, 2, 3, ..., 20\}$. We note that the distributions of both activating probability and influencing delay are orthogonal to the proposed methods.

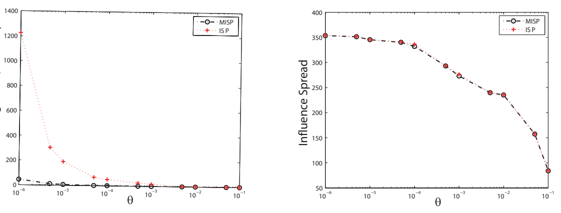
The threshold parameter for PMIA is set to $\frac{1}{320}$, which is the suggested value by [6]. We also try to run PMIA with other threshold values, which result in less influence spread.

Parameter $\theta$ controls the number of Influence Spreading Paths for MISP and ISP. Intuitively, smaller value of $\theta$ results in larger number of Influence Spreading Paths used by MISP and ISP, and thus should achieve larger influence spread. However, on the other hand, smaller value of $\theta$ incurs a larger

TABLE II
Statistics of Four Social Networks

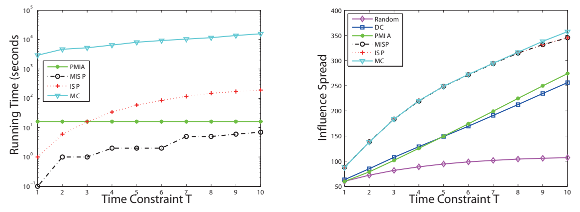
| Networks | Wiki | Epinions | Slashdot | LiveJournal |
|---|---|---|---|---|
| **Node Number** | $7,115$ | $75K$ | $82K$ | $4.8M$ |
| **Edge Number** | $103K$ | $508K$ | $948K$ | $68.9M$ |
| **Clustering Coefficient** | 0.2089 | 0.2283 | 0.0617 | 0.3123 |

[2]http://snap.stanford.edu/data/index.html



(a) Running Time (b) Influence Spread

Fig. 3. The results of running time and influence spread on Wiki with different $\theta$ ($T = 10, K = 50$).



(a) Running Time (b) Influence Spread

Fig. 6. The results of running time and influence spread on Wiki with different $T$ ($K = 50$).

amount of running time. So there exists a tradeoff between influence spread and running time, which is tunable by $\theta$.

To investigate this tradeoff and select an optimal value of $\theta$, we run MISP and ISP with different values of $\theta$. The running time and influence spread for different $\theta$ on Wiki dataset with $T = 10$, $K = 50$ are depicted in Figure 3. Note that results for other datasets and/or different values of $T$ and $K$ are similar, which are not included in this paper due to the limited space. Not surprisingly, Figure 3 shows that smaller value of $\theta$ achieves larger influence spread but consumes more running time for both MISP and ISP methods. As MISP and ISP achieve relatively large influence spread and short running time with $\theta = 10^{-5}$, $\theta$ is set to $10^{-5}$ for MISP and ISP in the rest of the experimentation,

*4) Measurement:* For time constrained social influence maximization problem, a critical performance metric is the number of nodes influenced by the selected seed set within a given time. As the time constrained influence maximization problem is *NP-hard*, we are not able to get the result in polynomial time. Thus we apply $20,000$ Monte Carlo simulations with seed set selected by each evaluated method, and the average influenced node number is used as the influence spread of the seed set. We also measure the running time and memory needed for each method. Furthermore, we will analyze the impacts of different values of $T$ on the time constrained influence maximization problem.

*5) Experimental Platform:* All algorithms are implemented in C++ language, and compiled by gcc 4.4.3 on a Linux server with an 8-core Intel Xeon 3.0 GHz CPU and 12 GB memory.

## B. Experimental Results

In this section, we present the experimental results of the proposed methods on four real world social networks.

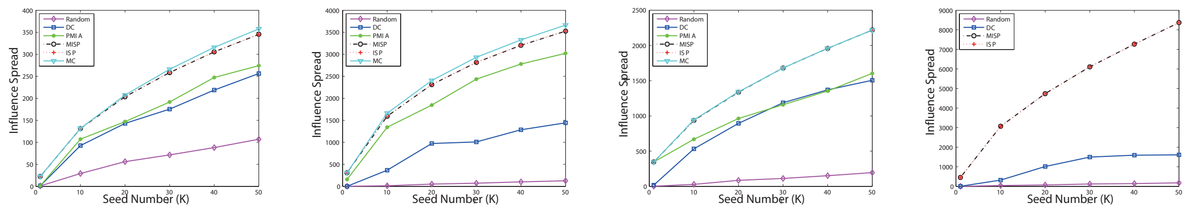
*1) Influence Spread:* All six methods indicated in Section V-A2 are evaluated over datasets Wiki, Epinions, and Slashdot. However, PMIA is not evaluated on the LiveJournal dataset as the memory needed for PMIA exceeds $12GB$, which is the amount of memory on the testing server. We cannot obtain the result for MC method on LiveJournal dataset after running it for two days.

Figure 4 shows the results of influence spread over the four datasets with $T = 10$ for different $K$ values. It shows that both ISP and MISP methods achieve similar influence spread as the computationally expensive greedy algorithm MC, which verifies the effectiveness of Influence Spreading Path based methods. We also observe that ISP and MISP achieve similar influence spread although MISP is an approximate version of ISP. As expected, a larger number of seed nodes lead to larger influence spread for all evaluated methods, and randomly selected seed set achieves very poor performance.

Among the algorithms for conventional influence maximization problem, PMIA performs the best, but it achieves considerably lower influence spread than do MISP, ISP and MC, which demonstrates that methods for conventional influence maximization problem do not work for the time constrained version.

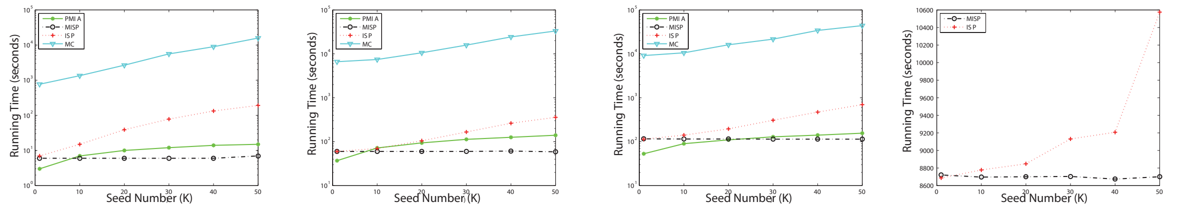
*2) Running Time:* Figure 5 shows the running time of different methods for each dataset with $T = 10$. As the running time for Random and DC is trivial, we do not include them to make the figure more distinguishable. When $K = 1$, ISP and MISP have similar running time, which is about two orders of magnitude faster than MC. The running time of ISP and MC increases as $K$ increases, while the running time of MISP almost remains constant for different values of $K$. These observations are due to the fact that MISP follows the same way as ISP to select the first seed node (by Influence Spreading Path), but employs a faster marginal influence spread estimation mechanism to select the rest seed nodes. The time needed by MISP is dominated by selecting the first seed node, and thus the total running time of MISP is almost constant for different values of $K$. We note that MISP is nearly three orders of magnitude faster than MC when $K = 50$. For small values of $K$, PMIA runs faster than all other methods except DC and Random, which are not depicted, on the three datasets where PMIA can return results. However, for a large $K$ ($K > 10$ for Wiki and Epinions, $K > 20$ for Slashdot), MISP is faster than PMIA.

*3) Memory Usage:* Table V-B3 shows the peak memory usage of each methods for different datasets with $T = 10$. We find that Random, MISP and MC always need the same amount of memory, which is mainly occupied by the social network data. For the two Influence Spreading Path based methods, the memory consumption of ISP increases as $K$ increases, while the memory needed by MISP remains constant. PMIA consumes the largest amount of memory, which renders it inapplicable to social networks of large scale (i.e., LiveJournal).



(a) Wiki (b) Epinions (c) Slashdot (d) LiveJournal

Fig. 4. The results of influence spread on four real world social networks ($T = 10$).



(a) Wiki (b) Epinions (c) Slashdot (d) LiveJournal

Fig. 5. The results of running time on four real world social networks ($T = 10$).

TABLE III
MEMORY USAGE (MB) OF DIFFERENT METHODS ($T = 10$)

| $K =$ | 1 | 10 | 20 | 30 | 40 | 50 |
|---|---|---|---|---|---|---|
| Wiki | | | | | | |
| Random | 13 | 13 | 13 | 13 | 13 | 13 |
| DC | 19 | 19 | 19 | 19 | 19 | 19 |
| PMIA | 19 | 19 | 19 | 20 | 20 | 20 |
| MISP | 13 | 13 | 13 | 13 | 13 | 13 |
| ISP | 13 | 19 | 20 | 21 | 21 | 21 |
| MC | 13 | 13 | 13 | 13 | 13 | 13 |
| Epinions | | | | | | |
| Random | 51 | 51 | 51 | 51 | 51 | 51 |
| DC | 74 | 74 | 74 | 74 | 74 | 74 |
| PMIA | 145 | 146 | 147 | 147 | 148 | 149 |
| MISP | 51 | 51 | 51 | 51 | 51 | 51 |
| ISP | 51 | 57 | 74 | 78 | 82 | 83 |
| MC | 51 | 51 | 51 | 51 | 51 | 51 |
| Slashdot | | | | | | |
| Random | 85 | 85 | 85 | 85 | 85 | 85 |
| DC | 119 | 119 | 119 | 119 | 119 | 119 |
| PMIA | 186 | 187 | 188 | 188 | 189 | 190 |
| MISP | 85 | 85 | 85 | 85 | 85 | 85 |
| ISP | 85 | 99 | 106 | 138 | 142 | 147 |
| MC | 85 | 85 | 85 | 85 | 85 | 85 |
| LiveJournal | | | | | | |
| Random | 5785 | 5785 | 5785 | 5785 | 5785 | 5785 |
| DC | 8358 | 8358 | 8358 | 8358 | 8358 | 8358 |
| PMIA | N.A | N.A | N.A | N.A | N.A | N.A |
| MISP | 5785 | 5785 | 5785 | 5785 | 5785 | 5785 |
| ISP | 5785 | 5785 | 5785 | 5785 | 5785 | 5785 |
| MC | N.A | N.A | N.A | N.A | N.A | N.A |

*4) Effect of Different Values of $T$:* To investigate whether different time constraint values of $T$ result in different seed sets maximizing time constrained influence, we run MC with Wiki, Slashdot and Epinions datasets for $T \in \{1, 2, ..., 10\}$ (as indicated in the previous sections, we cannot run MC with LiveJournal). Table IV depicts the overlaps of seed sets returned by MC for different values of $T$ with $K = 50$. For example, value 34 at row $T = 1$ and column $T = 4$ (Wiki) is the number of common nodes for the two T values. We find that seed sets maximizing influence spread with different time constraints differ significantly. For example, seed sets for $T = 1$ and $T = 10$ have only 22, 17 and 11 out of 50 nodes in common for Wiki, Epinions and Slashdot, respectively. We argue that time constraint plays an important role in influence maximization problem, and the set of nodes maximizing influence spread before a given time do not necessarily maximize that for a different time constraint.

To investigate how the value of $T$ affects the running time needed and influence spread achieved by different methods, we depict the running time and influence spread for different $T$ on Wiki dataset with $K = 50$ in Figure 6. Note that results for other datasets and/or different values of $K$ are similar, which are not included in this paper due to the limited space. As the running time for Random and DC is trivial, to make the figure more distinguishable, Random and DC methods are excluded from Figure 6(a). From Figure 6(a), we find that the running time of MISP, ISP and MC increases as $T$ increases, while that of PMIA remains constant. MISP achieves much less running time than MC and ISP. Again, MC needs the largest amount of running time among all methods. From Figure 6(b), we find that all methods achieve more influence spread as $T$ increases. This is due to the fact that a larger value of $T$ poses less restriction on time slots during which influence spread is counted. Again, MC, ISP and MISP achieve similar influence spread, which is much more than that of other methods.

## C. Summary and Discussion

From the experimental results, we find that time constraint plays an important role in influence maximization problem. We find that straightforward methods (Random and DC) are not suitable for the time constrained influence maximization problem, as they achieve poor influence spread. PMIA, a state-of-the-art solution for the conventional influence maximization problem, achieves much less time constrained influence spread than do MC, ISP and MISP. Another drawback of applying PMIA to maximize time constrained influence is its large memory consumption, which makes it unsuitable for large scale social networks. By investigating the effect of different values of $T$, we find that the set of nodes maximizing influence

TABLE IV
THE OVERLAPS OF SEED SETS RETURNED BY MC WITH DIFFERENT $T$ ($K = 50$)

| $T$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Wiki | | | | | | | | | | |
| 1 | 50 | 43 | 36 | 34 | 32 | 30 | 28 | 24 | 23 | 22 |
| 2 | | 50 | 41 | 39 | 37 | 35 | 33 | 29 | 27 | 26 |
| 3 | | | 50 | 46 | 44 | 42 | 39 | 33 | 31 | 28 |
| 4 | | | | 50 | 48 | 46 | 42 | 36 | 34 | 31 |
| 5 | | | | | 50 | 47 | 43 | 37 | 35 | 32 |
| 6 | | | | | | 50 | 46 | 40 | 38 | 35 |
| 7 | | | | | | | 50 | 44 | 42 | 39 |
| 8 | | | | | | | | 50 | 48 | 45 |
| 9 | | | | | | | | | 50 | 46 |
| 10 | | | | | | | | | | 50 |
| Epinions | | | | | | | | | | |
| 1 | 50 | 44 | 34 | 28 | 22 | 18 | 18 | 18 | 18 | 17 |
| 2 | | 50 | 40 | 34 | 28 | 23 | 22 | 22 | 21 | 20 |
| 3 | | | 50 | 43 | 37 | 31 | 30 | 30 | 29 | 27 |
| 4 | | | | 50 | 44 | 38 | 36 | 36 | 34 | 32 |
| 5 | | | | | 50 | 44 | 40 | 39 | 37 | 35 |
| 6 | | | | | | 50 | 45 | 43 | 41 | 39 |
| 7 | | | | | | | 50 | 48 | 46 | 44 |
| 8 | | | | | | | | 50 | 48 | 46 |
| 9 | | | | | | | | | 50 | 48 |
| 10 | | | | | | | | | | 50 |
| Slashdot | | | | | | | | | | |
| 1 | 50 | 42 | 36 | 28 | 23 | 19 | 15 | 13 | 12 | 11 |
| 2 | | 50 | 44 | 36 | 31 | 27 | 23 | 20 | 20 | 16 |
| 3 | | | 50 | 42 | 37 | 31 | 25 | 22 | 22 | 18 |
| 4 | | | | 50 | 45 | 39 | 33 | 30 | 30 | 26 |
| 5 | | | | | 50 | 43 | 36 | 33 | 33 | 29 |
| 6 | | | | | | 50 | 43 | 40 | 39 | 35 |
| 7 | | | | | | | 50 | 47 | 46 | 42 |
| 8 | | | | | | | | 50 | 48 | 45 |
| 9 | | | | | | | | | 50 | 46 |
| 10 | | | | | | | | | | 50 |

spread before a given time do not necessarily maximize that for a different time constraint, which shows that time constraint plays an important role in influence maximization problem.

Influence Spreading Path based methods (ISP and MISP) run much faster than the expensive MC, and achieve similar influence spread. Overall, MISP is the all-round winner for the time constrained influence maximization problem, as it achieves influence spread very close to MC and is multiple orders of magnitude faster than MC. Other nice properties of MISP include that its running time almost remains constant as $K$ increases, and consumes the same amount of memory as Random and MC, which is mainly the space needed to store the social network.

## VI. CONCLUSION

In this paper, we propose a new problem of the time constrained influence maximization in social networks based on a Latency Aware Independent Cascade model. We prove its *NP-hardness*, and develop a simulation based greedy algorithm with performance guarantees to solve the problem. However, the simulation based implementation of the greedy algorithm is quite expensive, and is not suitable for large social networks. We propose to use Influence Spreading Paths to quickly and effectively approximate the time constrained influence spread for a given seed set, which is the expensive part of the greedy algorithm. Further, by employing faster marginal influence spread calculating methods, we propose MISP to improve the speed of ISP. Experimental results on four public available datasets show that MISP is the fastest and multiple orders of magnitude faster than simulation based greedy algorithm MC while achieving similar time constrained influence spread. Other nice properties of MISP include that its running time almost remains constant as $K$ increases, and its memory usage is very efficient.

This work suggests a number of promising directions for future work. First, as the size of social networks is growing fast, it is unlikely to load a huge social network into memory. Thus developing distributed version of time constrained influence maximization algorithms is of great value. Second, besides IC model, there exist other influence propagation models, such as Linear Threshold model. How to solve time constrained influence maximization problem under these models is an interesting direction. Finally, as the proposed methods deal with discrete time only, we will study the extension of these methods for the case of continuous time in the future.

## ACKNOWLEDGMENT

The authors would like to thank Dr. Wei Chen for providing us the implementations of DC and PMIA.

## REFERENCES

[1] P. Domingos and M. Richardson, "Mining the network value of customers," in KDD, pp. 57–66, 2001.
[2] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in KDD, pp. 61–70, 2002.
[3] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in KDD, pp. 137–146, 2003.
[4] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in KDD, pp. 420–429, 2007.
[5] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in KDD, pp. 199–208, 2009.
[6] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in KDD, pp. 1029–1038, 2010.
[7] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-K influential nodes in mobile social networks," in KDD, pp. 1039–1048, 2010.
[8] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "A data-based approach to social influence maximization," in PVLDB, vol. 5, pp. 73–84, 2011.
[9] F. Bass, "A new product growth model for consumer durables," in Management Science, vol. 15, pp. 215–227, 1969.
[10] V. Mahajan, E. Muller, and F. M. Bass, "New Product Diffusion Models in Marketing: A Review and Directions for Research," in The Journal of Marketing, vol. 54, no. 1, pp. 1–26, 1990.
[11] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "Learning influence probabilities in social networks," in WSDM, pp. 241–250, 2010.
[12] M. G. Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," in KDD, pp. 1019–1028, 2010.
[13] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the Temporal Dynamics of Diffusion Networks," in ICML, pp. 561–568, 2011.
[14] K. Saito, M. Kimura, K. Ohara, and H. Motoda, "Selecting Information Diffusion Models over Social Networks for Behavioral Analysis," in PKDD, pp. 180–195, 2010.
[15] K. Saito, K. Ohara, Y. Yamagishi, M. Kimura, and H. Motoda, "Learning diffusion probability based on node attributes in social networks," in ISMIS, pp. 153–162, 2011.
[16] W. Chen, W. Lu, and N. Zhang, "Time-critical influence maximization in social networks with time-delayed diffusion process," in AAAI, pp. 1–5, 2012.
[17] M. Kimura and K. Saito, "Tractable models for information diffusion in social networks," in PKDD, pp. 259–271, 2006.
[18] J. Goldenberg, B. Libai, and E. Muller, "Talk of the Network: A Complex Systems Look at the Underlying Process of Word-of-Mouth," in Marketing Letters, pp. 211–223, 2001.
[19] M. Kimura, K. Saito, and H. Motoda, "Blocking links to minimize contamination spread in a social network," in TKDD, vol. 3, pp. 1–22, 2009.
[20] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," in Mathematical Programming, vol. 14, no. 1, pp. 265–294, 1978.