Influence Maximization with Novelty Decay in Social Networks

Shanshan Feng,[1] **Xuefeng Chen,**[2] **Gao Cong,**[3]
[1] Interdisciplinary Graduate School, Nanyang Technological University, Singapore, sfeng003@e.ntu.edu.sg
[2] University of Electronic Science and Technology of China, China, cxflovechina@gmail.com
[3] School of Computer Engineering, Nanyang Technological University, Singapore, gaocong@ntu.edu.sg

Yifeng Zeng,[4] **Yeow Meng Chee,**[5] **Yanping Xiang**[2]
[4] School of Computing, Teesside University, UK, Y.Zeng@tees.ac.uk
[5] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore, ymchee@ntu.edu.sg
[2] University of Electronic Science and Technology of China, China, xiangyanping@gmail.com

Abstract

Influence maximization problem is to find a set of seed nodes in a social network such that their influence spread is maximized under certain propagation models. A few algorithms have been proposed for solving this problem. However, they have not considered the impact of *novelty decay* on influence propagation, i.e., repeated exposures will have diminishing influence on users. In this paper, we consider the problem of influence maximization with novelty decay (IMND). We investigate the effect of novelty decay on influence propagation on real-life datasets and formulate the IMND problem. We further analyze the problem properties and propose an influence estimation technique. We demonstrate the performance of our algorithms on four social networks.

Introduction

As a fundamental research problem in social networks, *influence maximization* problem has attracted significant attention (Kempe, Kleinberg, and va Tardos 2003). It selects a set of K nodes in order to maximize the propagation of ideas, opinions, etc. in social networks. The influence maximization problem has many real-world applications. For example, a marketing campaign may target a small set of influential individuals and expect that the selected users would generate the largest influence coverage in the market.

It has been observed that repeated exposures of an individual to an idea may have diminishing influence on the individual (Ver Steeg, Ghosh, and Lerman 2011). For example, in Twitter, a user is more likely to retweet a tweet message for the first time that the user reads the message than the subsequent exposures to the message. The chance that a user retweets a message usually diminishes with the number of repeated exposures to the message. We call the phenomenon *novelty decay*. Intuitively, people are less likely to become spreaders of repeated information.

To the best of our knowledge, the novelty decay phenomenon has not been considered by existing studies on the influence maximization problem. We show one example of influence propagation with novelty decay in Figure 1.

Four users are linked and the directed edges indicate the influence of one user over another. Each edge is associated with two values, namely influence probability P and expected influence delay time T. For example, user V_1 influences V_3 with a probability of 0.7 in 2 time units. Given the seed set $\{V_1, V_2\}$, the probability that V_3 is activated by the seed set can be computed regularly without considering the novelty decay, e.g., $0.1 + (1 - 0.1) \times 0.7$, where 0.1 (resp. $(1 - 0.1) \times 0.7$) is the probability that V_3 gets activated by V_2 (resp. V_1). As V_3 is influenced by both V_1 and V_2 probably in a certain order, the novelty decay shall be considered into the influence propagation. Consequently, the probability of V_3 being activated by V_1 will diminish (less than $(1 - 0.1) \times 0.7$) if V_2 first attempts to activate V_3.

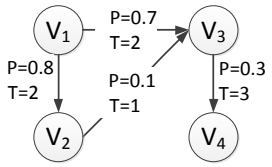


Figure 1: A social network with influence probabilities (P) and delays (T) on the directed edges.

In this paper, we analyze the effect of novelty decay on influence propagation based on two real-life datasets. For a further examination, we develop a fitting function to characterize the effect. By doing this, we are able to integrate the novelty decay factor into an influence propagation model such as independent cascade (IC), which is widely used in the literature (Chen, Wang, and Yang 2009; Chen, Lu, and Zhang 2012; Liu et al. 2012).

Differing from conventional influence propagation models, the new influence function becomes neither *monotone* nor *submodular*. This renders inapplicable the greedy algorithm with the CELF optimization (Leskovec et al. 2007) that is adopted by nearly all the existing influence maximization algorithms. We then improve the adapted version of U-Greedy algorithm (Lu and Lakshmanan 2012) by prun-

ing low-influential nodes in a dynamic way. As the influence propagates differently with the delay time between every pair of nodes, computing influence spread becomes complex. We develop a propagation path based algorithm to estimate the influence spread of seed nodes. Experimental results show that our algorithms can achieve large influence spread efficiently.

Related Work

Novelty Decay

Steeg et al. (Ver Steeg, Ghosh, and Lerman 2011) observe that multiple exposures to a story only marginally increase the probability of voting for it in a social network, and thus people are less likely to become spreaders of repeated information. Moreover, they find that the real influence generated by multiple exposures is much smaller than that computed by the IC model. However, they do not consider any computational model for studying the novelty decay factor in the context of influence maximization. Findings related to the novelty decay are also reported in other work. For example, the novelty within groups decays and attention to novel items fades over time (Wu and Huberman 2007).

Influence Maximization

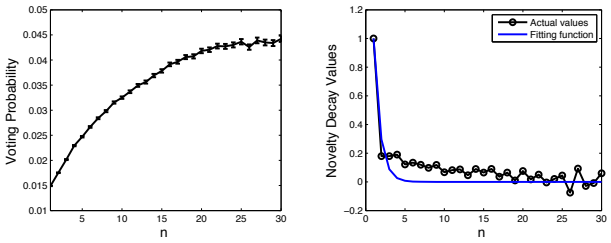
Influence maximization problem is formulated as a discrete optimization problem and its *NP-hardness* is proved by Kempe et al. (Kempe, Kleinberg, and va Tardos 2003). They develop a greedy algorithm framework, which iteratively selects the node with the largest marginal influence into the seed set until the number of seed nodes is reached.

Kempe et al. (Kempe, Kleinberg, and va Tardos 2003) introduce the IC model for influence propagation. Recently, time factor is incorporated into the IC model in the time constrained influence maximization problem (Chen, Lu, and Zhang 2012; Liu et al. 2012).

The problem of computing influence spread is *#P-hard* (Chen, Wang, and Wang 2010). A widely used baseline method for computing influence spread is based on Monte Carlo (MC) simulation (Kempe, Kleinberg, and va Tardos 2003). However, the MC simulation based algorithm is time consuming and not scalable in large social networks. Several heuristic algorithms have appeared. For example, maximum influence arborescence algorithm (Chen, Wang, and Wang 2010) employs the combination of the maximum influence paths to estimate influence spread. The independent paths based techniques (Liu et al. 2012; Kim, Kim, and Yu 2013) sum up influence of a limited number of paths that can be calculated independently from seed nodes to other nodes.

Novelty Decay in Influence Propagation

Inspired by the aforementioned findings on the novelty decay , we formalize the computation of the novelty decay and further confirm its effect on two publicly available datasets. With the novelty decay function, we proceed to formulate the influence maximization problem with the novelty decay.



(a) The voting probability of a user after n friends have voted. (b) Actual novelty decay values and its fitting function.

Figure 2: Novelty Decay on Digg.

Novelty Decay Function

For the sake of clarity, we assume that a user is exposed to an event n times if n friends of the user have been influenced. Formally, let TP_n be the probability that a user is influenced after n friends of the user get influenced, and p_n be the probability that a user gets influenced after the n^{th} friend of the user is influenced. We model the relationship between TP_n and TP_{n-1} in Eq. 1.

$$TP_n = TP_{n-1} + (1 - TP_{n-1}) \times p_n \quad (1)$$

Then we compute p_n in Eq. 2

$$p_n = (TP_n - TP_{n-1})/(1 - TP_{n-1}) \quad (2)$$

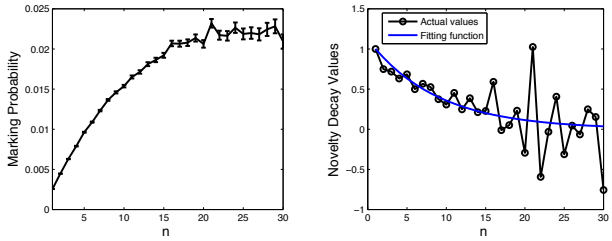
In order to formalize the novelty decay function, $f(n)$, we isolate the novelty decay factor from p_n. Specifically, we compute $f(n) = p_n/p_1$, where $p_1 = TP_1$ is the average probability to be influenced when users get influenced at the first time. We further employ the exponential function, $f(n)$=γ^{n-1}, as a general form of the novelty decay function and apply *least squares* approach to estimate its parameter γ. We show the development of the novelty decay functions in both Digg and Flickr datasets.

Digg Dataset contains information about stories promoted to the front page of Digg (digg.com) in June 2009 (Lerman and Ghosh 2010). The network has 279,634 nodes and 1,731,658 edges. If user u lists user v as a friend, u can see v's activities. The dataset also lists the Digg-votes, each of which records users' voting on a particular story and the voting time. It contains 3,018,197 votes from 139,409 distinct users on 3,553 popular stories.

Figure 2(a) demonstrates that the voting probability of a user over a story, TP_n, approaches a saturation point when a sufficient number of her friends (n >25) have voted for the story. In Figure 2(b), the actual novelty diminishes with repeated exposures. As illustrated by the blue line in Figure 2(b), the best fitting function is $f(n) = 0.2969^{n-1}$, which leads to the smallest sum of squared errors (SSE=0.1941).

Flickr Dataset contains a friendship graph and a list of favorite marking records from Flickr (www.flickr.com) (Cha, Mislove, and Gummadi 2009). If a user u lists v as its friend, u can see the activity (marking photos as favorites) of v. To study how influence propagates through the Flickr social network, we consider active users (having at least 5 markings) and active photos (having been marked by at least 100 users). There are 222,038 active users connected by 14,727,116 links and 3,125 active photos.

In Figure 3(a), the marking probability, $TP(n)$, increases with n in the beginning and then becomes stable around



(a) The marking probability of a user after n friends have voted. (b) Actual novelty decay values and its fitting function.

Figure 3: Novelty Decay on Flickr.

$n >$23. Figure 3(b) confirms the effect of novelty decay and shows the best fitting function $f(n) = 0.8918^{n-1}$ with the smallest SSE (SSE=2.7570).

IC Model with Novelty Decay

In the IC model with time delay, every node has two states: *active* and *inactive*. It is allowed to switch from inactive to active states, but not vice versa. Each edge is associated with two parameters, namely influence probability $\mathcal{P}_{uv}$ and expected influencing delay time T_{uv}. We augment the IC model with the novelty decay, denoted as $\mathsf{IC_{ND}}$, for each node in a social network.

Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a seed set $S \subseteq \mathcal{V}$ and a novelty decay function $f(n)$, the $\mathsf{IC_{ND}}$ model works as follows. Let A_t be the set of nodes activated at time $t \geq 0$, and $A_0 = S$. Every node $u \in A_t$ has a single chance to activate its out-neighbors that are inactive at time $t + T_{uv}$. Node u activates v with the probability $\mathcal{P}_{uv} \times f(n)$, where n is the number of exposures v has received. An exposure represents a chance that an active node intends to activate an inactive node. The influence propagation process terminates if and only if there is no any exposure. The number of all active nodes is denoted as $\sigma(S) = \sum_{t=0}^{\infty} |A_t|$.

Problem Definition and Properties

Based on the proposed $\mathsf{IC_{ND}}$ model, we formulate the problem of influence maximization with novelty decay (IMND).

Definition 1. ***(Influence Maximization with Novelty Decay)*** *Given a social network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a novelty decay function $f(n)$ and a positive integer K, find a seed set $S \subseteq \mathcal{V}$ that maximizes the expected number of nodes influenced by S under the $\mathsf{IC_{ND}}$ model.*

$$S = argmax_{S \subseteq \mathcal{V}, |S| \leq K} \{\sigma(S) | f(n)\}$$

The conventional influence maximization (IM) problem has been proved to be *NP-hard* (Kempe, Kleinberg, and va Tardos 2003). As it is a special case of IMND with $f(n) = 1$ for all n (i.e. there is no novelty decay), we get the hardness of IMND as follows.

Proposition 1. *The Influence Maximization Problem with Novelty Decay is NP-hard for the $\mathsf{IC_{ND}}$ model.*

Unlike the conventional IM problem, the influence function $\sigma(S)$ is non-monotone and non-submodular in IMND. For the proof purpose, we list the special cases when the novelty decay factor is considered in the example of social network (given $f(n) = 0.3^{n-1}$) in Figure 1.

Case 1: Non-monotonicity suppose that $S_1 = \{V_1\}$, $S_2 = \{V_1, V_2\}$ and $S_3 = \{V_1, V_2, V_3\}$, then $\sigma(S_1) = 2.7204$, $\sigma(S_2) = 2.3757$ and $\sigma(S_3) = 3.3$. Because $\sigma(S_3) > \sigma(S_1) > \sigma(S_2)$, thus $\sigma(S)$ is non-monotone.

Case 2: Non-submodularity suppose that $S_1 = \{V_1\}$, $S_2 = \{V_1, V_2\}$, then $\sigma(S_1 \cup \{V_3\}) - \sigma(S_1) = 0.3796$, and $\sigma(S_2 \cup \{V_3\}) - \sigma(S_2) = 0.9243$. As $S_1 \subseteq S_2$ and $\sigma(S_1 \cup \{V_3\}) - \sigma(S_1) < \sigma(S_2 \cup \{V_3\}) - \sigma(S_2)$, $\sigma(S)$ is non-submodular.

Proposition 2 summarizes the properties.

Proposition 2. *The influence function under $\mathsf{IC_{ND}}$ model is neither submodular nor monotone.*

Greedy Algorithm and Optimization

Due to the non-monotonicity and non-submodularity of the influence spread function under $\mathsf{IC_{ND}}$, the traditional greedy algorithm (Kempe, Kleinberg, and va Tardos 2003) becomes inapplicable. We resort to the U-Greedy algorithm (Lu and Lakshmanan 2012) that is developed for solving profit maximization problems (with the non-monotonicity and submodularity properties) in social networks. We further improve the algorithmic efficiency with an optimization.

R-Greedy Algorithm

The U-Greedy algorithm repeatedly adds nodes of the maximal *positive* marginal profit, and returns a seed set of any size that results in the maximum profit. Note that the IMND problem seeks for a seed set having not larger than K nodes ($|S| \leq K$). We choose the first K nodes each of which has the maximal marginal influence, and then pick the set of seed nodes with the largest influence spread. The adapted algorithm is called the plain restricted greedy algorithm (R-Greedy). Although the size of returned seed set could be smaller than K in a general case, the situation will seldom occur in practice due to the limited budget (where K is relatively small compared to the network size).

Dynamic Pruning Optimization

Let S_k (s_k) denote the set of selected seeds (the single node) at round k. The R-Greedy algorithm needs to compute the marginal influence increase for each node $u \in \mathcal{V} \backslash S_{k-1}$ and is invoked at each iteration of the algorithm to retrieve the k^{th} node s_k. To achieve better efficiency, we develop an optimization approach, namely dynamic pruning method (DP), that exploits the previous computation of influence spread to select potential seed nodes in the R-Greedy algorithm. The complete algorithm is shown in Algorithm 1.

Let Q_k store the checked nodes in round k, and elements in Q_k are in the form of tuple (u, Inf_k^u), where $Inf_k^u = \sigma((S_{k-1} \cup \{u\})$ denotes the influence after adding u into selected seed set S_{k-1}. In each round, DP checks all the candidate nodes in decreasing order of their influence. It terminates once the influence of individual nodes is below a dynamically maintained threshold, $maxMarInf$, which records the maximum marginal influence increase in round k (Line 5). Another key optimization technique is that if u has been checked in round $(k-1)$, we derive an upper bound of its marginal influence, i.e., $(Inf_{k-1}^u +$

Algorithm 1: R-Greedy Algorithm with DP

Input: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, T_{uv}, $\mathcal{P}_{uv}$, $f(.)$ and K
Output: S

1 $S \leftarrow \emptyset$, $S_0 \leftarrow \emptyset$, $\sigma(S) \leftarrow 0$, $\sigma(S_0) \leftarrow 0$, $s \leftarrow NULL$;
2 For every $v \in \mathcal{V}$, calculate $\sigma(\{v\})$ and insert $(v,\sigma(\{v\}))$ into Q_0 ;
3 **for** $k \leftarrow 1$ **to** K **do**
4 $maxMarInf \leftarrow -\infty$;
5 **for** *node* $u \in \mathcal{V} \setminus S_{k-1}$, $\sigma(\{u\}) \geq maxMarInf$ **do**
6 **if** $u \in Q_{k-1}$, *and* $(Inf^u_{k-1} + \sigma(\{s_{k-1}\})) - \sigma(S_{k-1}) < maxMarInf$ **then**
7 Continue
8 **else**
9 Calculate Inf^u_k and insert (u, Inf^u_k) into Q_k;
10 **if** $Inf^u_k - \sigma(S_{k-1}) > maxMarInf$ **then**
11 $maxMarInf \leftarrow Inf^u_k - \sigma(S_{k-1})$;
12 $s_k \leftarrow u$;
13 $S_k \leftarrow S_{k-1} \cup \{s_k\}$;
14 $\sigma(S_k) \leftarrow \sigma(S_{k-1}) + maxMarInf$
15 $S \leftarrow$ the S_k with maximum $\sigma(S_k)$ from $k = 1$ to $k = K$;
16 **return** S;

$\sigma(\{s_{k-1}\})) - \sigma(S_{k-1})$ (Line 6 and the correctness will be given in Proposition 3). If the upper bound is not larger than $maxMarInf$, the node is ignored (Line 7). Then the algorithm calculates the influence of $(S_{k-1} \cup \{u\})$, and stores the result into Q_k (Line 9). If the marginal influence is larger than $maxMarInf$, it updates $maxMarInf$ as well as s_k (Lines 10-12). Finally, we obtain the seed set S_k and its influence spread $\sigma(S_k)$ (Lines 13-14).

As the DP operation prunes nodes whose influence is smaller than $maxMarInf$, the algorithm still maintains the solution quality of the plain R-Greedy algorithm. We formally prove the property.

Proposition 3. *The DP optimization preserves the solution quality of the R-Greedy algorithm.*

PROOF. The influence function of the $\mathsf{IC_{ND}}$ model satisfies $\sigma(S_1 \cup S_2) \leq \sigma(S_1) + \sigma(S_2)$, $S_1, S_2 \subseteq \mathcal{V}$. For a node u, its marginal influence $MarInf^k(u) = \sigma(S_{k-1} \cup \{u\}) - \sigma(S_{k-1}) \leq \sigma(S_{k-1}) + \sigma(\{u\}) - \sigma(S_{k-1}) = \sigma(\{u\})$. If node u has been checked in prior round, $MarInf^k(u) = \sigma(S_{k-1} \cup \{u\}) - \sigma(S_{k-1}) = \sigma((S_{k-2} \cup \{s_{k-1}\}) \cup \{u\}) - \sigma(S_{k-1}) = \sigma((S_{k-2} \cup \{u\}) \cup \{s_{k-1}\}) - \sigma(S_{k-1}) \leq Inf^u_{k-1} + \sigma(\{s_{k-1}\}) - \sigma(S_{k-1})$. Therefore, if the upper bound of a node u ($\sigma(\{u\})$ or $Inf^u_{k-1} + \sigma(\{s_{k-1}\}) - \sigma(S_{k-1})$) is smaller than $maxMarInf$, it's safe for DP to prune this node. □

As we need to check all of the nodes in every round, the time complexity of the R-Greedy algorithm is $O(K|\mathcal{V}|T(\sigma(S)))$, where $T(\sigma(S))$ is the time for computing $\sigma(S)$. With the DP optimization, the number of checked nodes in every round is much smaller than $|\mathcal{V}|$.

Algorithms for Computing Influence Spread

With the improved R-Greedy algorithm, the remaining issue is to compute influence spread of seed nodes. For this purpose, we propose a propagation path based algorithm that overcomes the inefficiency of simulation-based techniques.

Simulation Based Algorithm

Monte Carlo simulatation-based algorithms are widely used as baselines in the study of influence maximization (Kempe, Kleinberg, and va Tardos 2003). We adapt the algorithm to simulate the spreading process of $\mathsf{IC_{ND}}$ model by considering both the spread delay time and the novelty decay effect in the model. To obtain the average influence spread value, we need to conduct a large number of simulations. Overall the time complexity of the simulation-based approach (MC), together with Algorithm 1, is $O(K|\mathcal{V}|R(|\mathcal{V}| + |\mathcal{E}|))$, where R is the number of simulations generally set at 20,000.

Propagation Path Based Algorithm

The simulation-based algorithm is time-consuming and not suitable for large social networks. We develop a propagation path based algorithm to efficiently estimate influence spread.

Propagation Path with Novelty Decay Given a seed set $S \subseteq \mathcal{V}$, the expected influence spread $\sigma(S) = \sum_{u \in \mathcal{V}} \mathcal{AP}_S(u)$, where $\mathcal{AP}_S(u)$ is the probability of u being activated by S. To estimate $\mathcal{AP}_S(u)$, we define a propagation path with novelty decay ($\mathsf{PP_{ND}}$) below.

Definition 2. ***(Propagation Path with Novelty Decay)*** *Given a seed set S and a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, a path $h = (u_1 \xrightarrow{e_1} u_2 \xrightarrow{e_2} u_3 \ldots \xrightarrow{e_{k-1}} u_k)$ in graph $\mathcal{G}$ is a propagation path with novelty decay ($\mathsf{PP_{ND}}$), if and only if $u_1 \in S$ and $u_i \notin S$ for $i \neq 1$, where $k > 1$.*

As a node cannot be activated more than once, a $\mathsf{PP_{ND}}$ path does not contain duplicate nodes. Its length is $Len(h) = \sum_{i=1}^{i=k-1} T_{e_i}$ while the probability can be computed as $\prod_{i=1}^{i=k-1} \mathcal{P}(e_i) \times \hat{E}(\tau^h(u_{i+1}))$, where $\hat{E}(\tau^h(u_{i+1}))$ is the expected novelty decay value for h on u_{i+1}. Note that $\hat{E}(\tau^h(u))$ depends on the order of all paths ending at u.

Computing $\hat{E}(\tau^h(u))$ A $\mathsf{PP_{ND}}$ path has two states, *connected* and *blocked*. The path is connected if it successfully activates u_1, ..., u_{k-1}; otherwise it is blocked. The path has a chance to activate its ending node iff it is connected. We denote the connected probability as $\mathcal{P}_{con} = \prod_{i=1}^{i=k-2} \mathcal{P}(e_i) \times \tau^h(u_{i+1})$, and the blocked probability becomes $\mathcal{P}_{blo} = 1 - \mathcal{P}_{con}$. If the activation from h is the i^{th} exposure for u_k, h is ranked as i^{th} among all the paths ending at u_k. We next propose an expected novelty method to compute $\hat{E}(\tau^h(u))$.

Suppose that h_c is the c^{th} shortest $\mathsf{PP_{ND}}$ path of u, i.e., there are $c-1$ paths shorter than h_c. The shorter a path, the earlier it activates its ending node. To compute $\hat{E}(\tau^{h_c}(u))$, we need to consider all possible combinations of states (connected or blocked) of $c-1$ paths. For example, if h_1 and h_2 are the first two shortest paths to activate u, the computation of $\hat{E}(\tau^{h_3}(u))$ needs to consider 4 cases: $\hat{E}(\tau^{h_3}(u))$

$= \mathcal{P}_{blo}(h_1)\times\mathcal{P}_{blo}(h_2)\times f(1)+\mathcal{P}_{blo}(h_1)\times\mathcal{P}_{con}(h_2)\times f(2)+ \mathcal{P}_{con}(h_1)\times\mathcal{P}_{blo}(h_2)\times f(2)+\mathcal{P}_{con}(h_1)\times\mathcal{P}_{con}(h_2)\times f(3).$

Finding $\mathsf{PP}_{\mathsf{ND}}$ For a given seed set S, we use $\mathsf{PP}_{\mathsf{ND}}(u, S)$ to denote all $\mathsf{PP}_{\mathsf{ND}}$ paths from S to node u. It is obvious that each path provides a chance for S to activate u. Since the number of paths, $|\mathsf{PP}_{\mathsf{ND}}(u, S)|$, grows exponentially with the number of seed nodes, finding $\mathsf{PP}_{\mathsf{ND}}(u, S)$ is computationally expensive for a large S. We apply two restrictions to eliminate the $\mathsf{PP}_{\mathsf{ND}}$ paths that have small influence contribution. First, we prune the paths with probabilities smaller than a specified threshold $\theta > 0$. Second, we retain at most C shortest paths in $\mathsf{PP}_{\mathsf{ND}}(u, S)$, because a user is unlikely to be influenced after many exposures due to the novelty decay effect. The resulting $\mathsf{PP}_{\mathsf{ND}}$ s are denoted by $\mathsf{PP}_{\mathsf{ND}\theta,C}(u, S)$.

Finding $\mathsf{PP}_{\mathsf{ND}\theta,C}(S)$ aims to search at most C shortest paths for each destination node from multi-source nodes with a threshold restriction. This differs from the state-of-the-art algorithms (Yen 1971; Eppstein 1998) for *K shortest path routing* because the algorithms focus on single source and the θ constraint cannot be easily incorporated. To fill this gap, we develop an adapted Dijkstra (AD) algorithm for finding $\mathsf{PP}_{\mathsf{ND}\theta,C}(S)$. Like Dijkstra algorithm, AD adopts a greedy search strategy to select the shortest path for the extension. To satisfy the θ constraint, AD only extends the path meeting the constraint in each iteration. Furthermore, we integrate the computation of path probabilities into AD.

In Algorithm 2, AD starts with initializing $\mathsf{PP}_{\mathsf{ND}\theta,C}(S)$, $Count(u)$ recording the number of paths on node u, and $PH_{con}(u)$ recording $\mathcal{P}_{con}$ of the found paths (Lines 1). To implement the greedy search strategy, AD initializes a min priority queue PH for storing the information of paths, each of which has T as the length of time and $path$ for the information of nodes and probabilities on edges (Line 2). At each iteration, AD chooses the path that has the minimum T in PH to extend (line 4). At lines 5–6, $\mathcal{P}_{wu} \times E(\hat{\tau_u}(P))$ is the probability u is activated by w on the $\mathsf{PP}_{\mathsf{ND}}$ P, then the probability that u is activated by P is $\mathcal{P}^P(u) = \mathcal{P}_{con} \times \mathcal{P}_{wu} \times E(\hat{\tau_u}(P))$. If P satisfies the restrictions, we insert it into $\mathsf{PP}_{\mathsf{ND}\theta,C}(S)$, update $Count(u)$, $PH_{con}(u)$, get new paths by extending it, and then insert the new paths into PH (Lines 8–12).

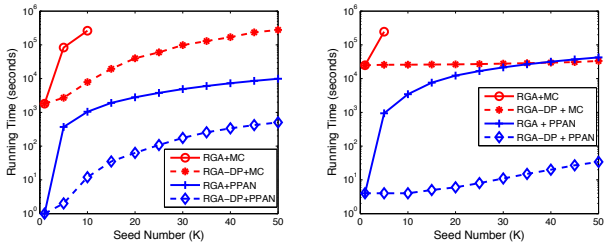
Computing $\sigma(S)$ After getting $\mathsf{PP}_{\mathsf{ND}\theta,C}(u, S)$, we compute the activation probability $\mathcal{AP}_S(u)$ (Line 14). All the paths ending at u are assumed to be independent following the previous work (Liu et al. 2012; Kim, Kim, and Yu 2013). Finally, the activating probabilities of all nodes are summed into $\sigma(S)$.

Let $N_{\theta,C} = max_{|S|\leq K}|\mathsf{PP}_{\mathsf{ND}\theta,C}(S)|$ be the maximum number of paths starting from S. The time complexity of finding $\mathsf{PP}_{\mathsf{ND}\theta,C}(S)$ is $O(N_{\theta,C}logN_{\theta,C})$. The calculation of $\sigma(S)$ takes $O(N_{\theta,C})$. Hence the total time complexity of Algorithm 2 is $O(N_{\theta,C}logN_{\theta,C})$. The combination of Algorithms 1 and Algorithms 2 takes $O(K|\mathcal{V}|N_{\theta,C}logN_{\theta,C})$. Since $N_{\theta,C}logN_{\theta,C} \ll R(|\mathcal{V}| + |\mathcal{E}|)$ in practice, the $\mathsf{PP}_{\mathsf{ND}}$ based solution is much faster than the simulation approach.

Algorithm 2: Computing $\sigma(S)$ based on $\mathsf{PP}_{\mathsf{ND}}$

Input: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, T_{uv}, S, $\mathcal{P}_{uv}$, $f(.)$, θ and C
Output: $\sigma(S)$

1 Initialize $\mathsf{PP}_{\mathsf{ND}\theta,C}(u, S) \leftarrow \emptyset$, $Count(u) \leftarrow 0$ and $PH_{con}(u) \leftarrow \emptyset$ for $u \in \mathcal{V}$;
2 Initialize a min priority queue PH, and enqueue $P = (0, 1, path = \{u\})$ for $u \in S$;
3 **while** $PH \neq \emptyset$ **do**
4 $P \leftarrow$ dequeue $(T, \mathcal{P}_{con}, path)$ from PH;
5 compute $\hat{E}(\tau^P(u))$ according to $PH_{con}(u)$;
6 $\mathcal{P}^P(u) \leftarrow \mathcal{P}_{con} \times \mathcal{P}_{wu} \times \hat{E}(\tau^P(u))$;
7 **if** *$Count(u) < C$ and $\mathcal{P}^P(u) > \theta$ and P is loopless* **then**
8 Insert $\mathcal{P}^P(u)$ into $\mathsf{PP}_{\mathsf{ND}\theta,C}(u, S)$;
9 $Count(u) \leftarrow Count(u) + 1$;
10 Insert $\mathcal{P}_{con}$ into $PH_{con}(u)$;
11 $\mathcal{P}_{con} \leftarrow \mathcal{P}^P(u)$;
12 enqueue $P = (T + T_{uv}, \mathcal{P}_{con}, path \cup \{v\})$ into PH for $(u, v) \in \mathcal{E}$;
13 **for** *every u with non-empty* $\mathsf{PP}_{\mathsf{ND}\theta,C}(u, S)$ **do**
14 $\mathcal{AP}_S(u) \leftarrow 1 - \prod_{P\in\mathsf{PP}_{\mathsf{ND}\theta,C}(S)}(1 - \mathcal{P}^P(u))$;
15 $\sigma(S) \leftarrow \sigma(S) + \mathcal{AP}_S(u)$;
16 return $\sigma(S)$;



(a) Wiki (b) NetPHY

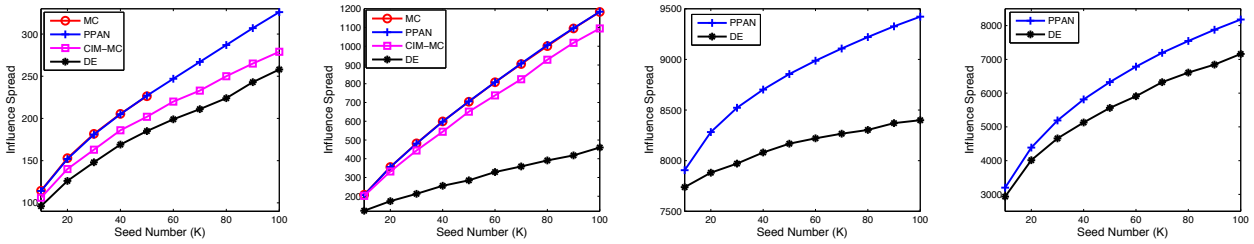
Figure 4: Performance of RGA and RGA-DP based algorithms.

Experiments

Experimental Setup

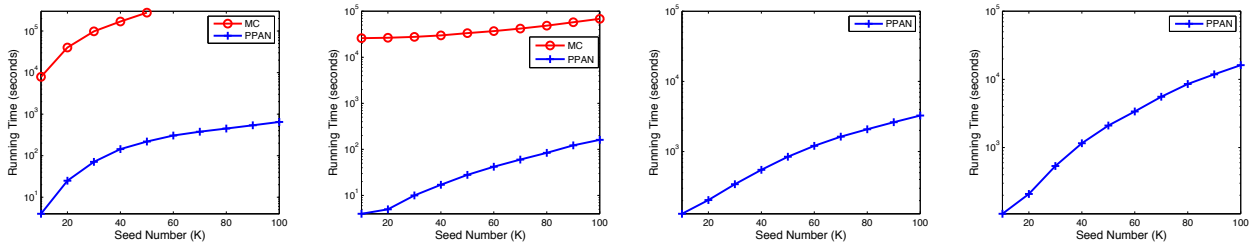
Datasets Apart from Digg and Flickr, two other real-world social networks are used in the experiments. Wiki is a voting network containing all the Wikipedia voting data from the inception of Wikipedia till January 2008. It has around 7,000 nodes and 103,000 edges. NetPHY is a collection network of papers, extracted from "Physics" sections in arXiv, and contains around 37,000 nodes and 181,000 edges.

Evaluated Methods We evaluate the plain R-Greedy algorithm **(RGA)** and the improved R-Greedy algorithm with DP (Algorithm 1) **(RGA-DP)**. Both algorithms use either the simulation based algorithm **(MC)** or the propagation path based algorithm (Algorithm 2) **(PPAN)** to compute influence spread. We then compare them to two conventional influence maximization algorithms: the classical MC based technique **CIM-MC** (Leskovec et al. 2007) and the degree based algorithm **DE** (choose K nodes with maximum degrees). The MC method is employed to compute the influence spread of the seed set returned by each method. All methods are implemented in C++ and experiments are conducted on a windows server with 6-core Intel(R) Xeon (R),



(a) Wiki (b) NetPHY (c) Digg (d) Flickr

Figure 5: The results of influence spread on four real world social networks.



(a) Wiki (b) NetPHY (c) Digg (d) Flickr

Figure 6: The results of running time on four real world social networks.

2.66 GHz CPU and 24 GB memory.

Parameter Setting We set the influence probability P_{uv} of u on v by the *weighted cascade policy* (Chen, Lu, and Zhang 2012; Liu et al. 2012), i.e., $P_{uv} = \frac{1}{indegree(v)}$, where $indegree(v)$ is indegree of node v. The expected influencing delay time T_{uv} of edge uv follows the *geometric delay distribution* (Chen, Lu, and Zhang 2012). The parameter for geometric distribution is set at $5/(outdegree(v) + 5)$. For simplicity, the maximum value of T_{uv} is 15. If the generated delay time $T_{uv} > 15$, T_{uv} is reset as a random integer from 1 to 15. We also try other distributions for T_{uv} including poisson distribution and uniform distribution, and the experiment results exhibit similar trend for the evaluated techniques. Empirically, threshold θ is set at $\theta = 0.001$ and number of paths C is set at $C = 5$, which achieves a satisfying tradeoff between influence spread and running time in our experiment. For Digg and Flickr, we use their actual novelty decay function $f(n)$ aforementioned. For wiki and NetPHY, we utilize the default exponential function $f(n) = 0.3^{n-1}$ and further examine different γ.

Selecting the R-Greedy Algorithms

We investigate the effect of DP optimization on the R-Greedy algorithm. Both MC and PPAN are used to compute influence spread. We terminate MC if it run over five days. Figure 4 shows the results on both Wiki and NetPHY since MC cannot finish for the large Digg and Flickr networks. Figure 4 demonstrates that RGA-DP is 1-2 orders of magnitude faster than RGA on both datasets. Hence we use RGA-DP for the rest of experiment if applicable. Note that DP optimization does not affect the solution quality of the RGA. Thus we do not compare their influence spread.

Performance of the Algorithms for Solving IMND

We evaluate all the algorithms in terms of influence spread and running time over four datasets. Since MC and CIM-MC are extremely computationally expensive, we cannot get their results on Digg and Flickr.

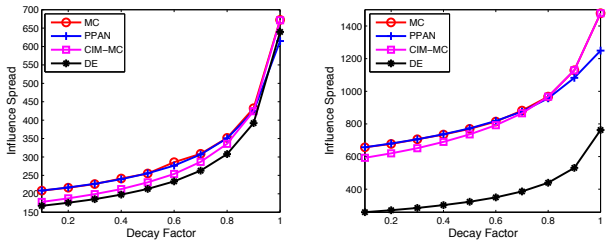
Influence Spread The influence spread reflects the quality of selected seed set. Figure 5 illustrates the influence spread for various K values. PPAN obtains similar influence spread as MC. It indicates that PPAN is an effective approximate method. Both CIM-MC and DE achieve consistently lower influence spread than do PPAN and MC. This demonstrates that conventional influence maximization techniques are not effective for solving the new IMND problem.

Running Time Figure 6 shows that the running time grows when K increases for the studied methods. PPAN is several orders of magnitude more efficient than MC. In summary, the combination of Algorithms 1 and 2 provides the best solution to the IMND problem.

Effect of $f(n)$ We further investigate the impact of γ on Wiki and NetPHY. Figure 7 shows the influence spread for various γ with $K = 50$. As expected, the influence spread grows as γ increases. This is due to the fact that if γ is small, the spread probability from one node to another is declined. Hence, the expected number of activated nodes becomes smaller. Additionally, the results of PPAN are consistently similar to that of MC for different γ. This verifies that PPAN is an effective approximate method for different $f(n)$.

Conclusion

We investigate the effect of novelty decay in social networks. With the augmented independent cascade propagation model, we formulate influence maximization problem



(a) Wiki (b) NetPHY

Figure 7: The results of influence spread for different γ.

with novelty decay. To solve the problem, we employ the R-Greedy algorithm and improve its performance through the dynamic pruning strategy. In addition, we propose an influence propagation path based algorithm to efficiently compute influence spread of seed nodes. Performance of our algorithms is demonstrated with extensive experiments.

Acknowledgements

Gao Cong was supported in part by a grant awarded by a Singapore MOE AcRF Tier 2 Grant (ARC30/12). ShanShan Feng would like to acknowledge the Ph.D. grant from Advanced Environmental Biotechnology Centre, Nanyang Environment and Water Research Institute, Interdisciplinary Graduate School, Nanyang Technological University, Singapore.

References

Cha, M.; Mislove, A.; and Gummadi, K. P. 2009. A measurement-driven analysis of information propagation in the flickr social network. In *WWW*, 721–730.

Chen, W.; Lu, W.; and Zhang, N. 2012. Time-critical influence maximization in social networks with time-delayed diffusion proces. In *AAAI*, 592–598.

Chen, W.; Wang, C.; and Wang, Y. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, 1029–1038.

Chen, W.; Wang, Y.; and Yang, S. 2009. Efficient influence maximization in social networks. In *KDD*, 199–208.

Eppstein, D. 1998. Finding the k shortest paths. *SIAM Journal on computing* 28(2):652–673.

Kempe, D.; Kleinberg, J. M.; and va Tardos. 2003. Maximizing the spread of influence through a social network. In *KDD*, 137–146.

Kim, J.; Kim, S.-K.; and Yu, H. 2013. Scalable and parallelizable processing of influence maximization for large-scale social networks. In *ICDE*, 266–277.

Lerman, K., and Ghosh, R. 2010. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *ICWSM*, 90–97.

Leskovec, J.; Krause, A.; Guestrin, C.; Faloutsos, C.; VanBriesen, J. M.; and Glance, N. S. 2007. Cost-effective outbreak detection in networks. In *KDD*, 420–429.

Liu, B.; Cong, G.; Xu, D.; and Zeng, Y. 2012. Time constrained influence maximization in social networks. In *ICDM*, 439–448.

Lu, W., and Lakshmanan, L. V. S. 2012. Profit maximization over social networks. In *ICDM*, 479–488.

Ver Steeg, G.; Ghosh, R.; and Lerman, K. 2011. What stops social epidemics. In *ICWSM*, 377–384.

Wu, F., and Huberman, B. A. 2007. Novelty and collective attention. *Proceedings of the National Academy of Sciences* 104(45):17599–17601.

Yen, J. Y. 1971. Finding the k shortest loopless paths in a network. *Management Science* 17(11):712–716.