Influence Maximization with Novelty Decay in Social Networks

Shanshan Feng1  Xuefeng Chen2  Gao Cong1  Yifeng Zeng3  Yeow Meng Chee1  Yanping Xiang2
1Nanyang Technological University, Singapore, \{sfeng003@, gaocong@, ymchee@\}ntu.edu.sg
2University of Electronic Science and Technology of China, China, \{cxlovechina, xiangyanping\}@gmail.com
3Teesside University, UK, Y.Zeng@tees.ac.uk

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 in 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 has attracted significant attention (Kempe, Kleinberg, and Tardos 2003; Chen, Lu, and Zhang 2012; Kim, Kim, and Yu 2013). It selects a set of K seed 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 (Leskovec, Adamic, and Huberman 2007; 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 V1 influences V3 with a probability of 0.7 in 2 time units. Given the seed set \{V1, V2\}, the probability that V3 is activated by the seed set can be computed regularly without considering the novelty decay, e.g., 0.1 + (1 − 0.1) × 0.7, where 0.1 (resp. (1 − 0.1) × 0.7) is the probability that V3 gets activated by V2 (resp. V1). As V3 is influenced by both V1 and V2 probably in a certain order, the novelty decay shall be considered into the influence propagation. Consequently, the probability of V3 being activated by V1 will diminish (less than (1 − 0.1) × 0.7) if V2 first attempts to activate V3.

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, Adamic, and Huberman 2007) that is adopted by nearly all the existing influence maximization algorithms. We then improve the adapted version of U-Greedy algorithm (Lu and Laksman 2012) by pruning 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

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved. Corresponding Author: Yanping Xiang.
achieve large influence spread efficiently.

Related Work

Novelty Decay
Stee 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. More importantly, 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), and novelty decay exists in recommendation systems (Leskovec, Adamic, and Huberman 2007).

Influence Maximization
Influence maximization problem is formulated as a discrete optimization problem and its NP-hardness is proved (Kempe, Kleinberg, and va Tardos 2003). Kempe et al. (Kempe, Kleinberg, and va Tardos 2003) 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 IC in the time constrained influence maximization problem (Chen, Lu, and Zhang 2012; Gomez-Rodriguez and Scholkopf 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 very time consuming and is not scalable in large social networks. Several heuristic algorithms have appeared. For example, the PMIA method (Chen, Wang, and Wang 2010) employs Maximum influence Arborescence(MIA), which is the combination of the maximum influence paths. The newly ISP 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.

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.

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

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, $f(n) = \gamma^{n-1}$, as a general form of the novelty decay function and apply least squares approach to estimate its parameter $\gamma$. 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 a user $v$ as its friend, $u$ can see the activity (marking photos as favorites) of $v$. To study how photos propagate 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 3125 active photos.

In Figure 3(a), the marking probability, $TP(n)$, increases with $n$ in the beginning and then becomes stable around $n > 23$. 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

\[\text{Voting Probability} \]
\[\text{Novelty Decay Values} \]

(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.

Then we compute $p_n$ in Eq. 2

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

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, $f(n) = \gamma^{n-1}$, as a general form of the novelty decay function and apply least squares approach to estimate its parameter $\gamma$. 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 a user $v$ as its friend, $u$ can see the activity (marking photos as favorites) of $v$. To study how photos propagate 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 3125 active photos.

In Figure 3(a), the marking probability, $TP(n)$, increases with $n$ in the beginning and then becomes stable around $n > 23$. 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
Problem Definition and Properties

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

\[
S = \arg\max_{S \subseteq V, |S| \leq K} \{ \sigma(S)f(n) \}
\]

The conventional influence maximization (IM) problem has been proved to be NP-hard (Kempe, Kleinberg, and 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 IC\(_{\text{ND}}\) 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 \( f(n) = 0.3^n \)) 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 \( \sigma(S_3) > \sigma(S_2) > \sigma(S_1) \), 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 IC\(_{\text{ND}}\) model is neither submodular nor monotone.

Greedy Algorithm and Optimization

Due to the non-monotonicity and non-submodularity of the influence spread function under IC\(_{\text{ND}}\), the traditional greedy algorithm (Kempe, Kleinberg, and Tardos 2003) becomes inapplicable. We resort to the U-Greedy algorithm (Lu and Lakshmanan 2012) that is developed for solving influence 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 influence until a full set of \( K \) nodes are reached. 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 shall 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 \( u \in V \setminus S_{k-1} \) 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 \( (u, Inf_u^{\text{S}_k}) \), where \( \text{Inf}_u^{\text{S}_k} = \sigma(S_{k-1} \cup \{u\}) \) 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., \( (\text{Inf}_u^{\text{S}_{k-1}} + \sigma(S_{k-1})) - \sigma(S_{k-1}) \) (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
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. $S \leftarrow \emptyset$, $S_0 \leftarrow \emptyset$, $\sigma(S) \leftarrow 0$, $\sigma(S_0) \leftarrow 0$, $s \leftarrow \text{NULL}$;
2. For every $v \in \mathcal{V}$, calculate $\sigma(\{v\})$ and insert $(v, \sigma(\{v\}))$ into $Q_v$;
3. for $k = 1$ to $K$ do
   for node $u \in \mathcal{V} \setminus S_{k-1}$, $\sigma(\{u\}) \geq \text{maxMarInf}$ do
      if $u \in Q_{k-1}$, and
         $(\text{Inf}_{u}^{f_{k-1}} + \sigma(\{s_{k-1}\})) - \sigma(S_{k-1}) < \text{maxMarInf}$
         then $\text{maxMarInf} \leftarrow \text{Inf}_{u}^{f_{k-1}} - \sigma(S_{k-1})$;
      else $\text{Calculate Inf}_{u}^{f_{k-1}}$ and insert $(u, \text{Inf}_{u}^{f_{k-1}})$ into $Q_k$;
      if $\text{Inf}_{u}^{f_{k}} - \sigma(S_{k-1}) > \text{maxMarInf}$ then
         $\text{maxMarInf} \leftarrow \text{Inf}_{u}^{f_{k}} - \sigma(S_{k-1})$;
      $S_k \leftarrow S_{k-1} \cup \{s_k\}$;
      $\sigma(S_k) \leftarrow \sigma(S_{k-1}) + \text{maxMarInf}$
   $S \leftarrow S_k$ with maximum $\sigma(S_k)$ from $k = 1$ to $k = K$;
4. return $S$;

$s_k$ (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 $\text{maxMarInf}$, 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 ICND 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 $\text{MarInf}^{f}(u) = \sigma(S_{k-1} \cup \{u\}) - \sigma(S_{k-1})$ is smaller than $\sigma(S_{k-1} \cup \{u\})$.

**Algorithm 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 simulation-based algorithms are widely used as baselines in the study of influence maximization (Kempe, Kleinberg, and va Tardos 2003). We adopt the algorithm to simulate the spreading process of ICND 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 Algorithms**

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 $\sigma(S) = \sum_{u \in \mathcal{V}} \mathcal{P}S(u)$, where $\mathcal{P}_S(u)$ is the probability of $u$ being activated by $S$. To estimate $\mathcal{P}_S(u)$, we define a propagation path with novelty decay (PPND) 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 $h = (u_1 \xrightarrow{c_1} u_2 \xrightarrow{c_2} u_3 \ldots \xrightarrow{c_{k-1}} u_k)$ in graph $\mathcal{G}$ is a propagation path with novelty decay $\mathcal{P}_{PPND}$, 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 PPND path does not contain duplicate nodes. Its length is $\text{Len}(h) = \sum_{i=1}^{k-1} T_{e_i}$, while the probability can be computed as $\prod_{i=1}^{k-1} \mathcal{P}(e_i) \times \hat{E}(\tau_h(u_i+1))$, 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 PPND path has two states, connected and blocked. The path is connected if it successfully activates $u_1, \ldots, u_{k-1}$; otherwise it is blocked. The path has a chance to activate its ending node if it is connected. We denote the connected probability as $\mathcal{P}_{\text{con}} = \prod_{i=1}^{k-2} \mathcal{P}(e_i) \times \tau_h(u_i+1)$, and the blocked probability becomes $\mathcal{P}_{\text{blo}} = 1 - \mathcal{P}_{\text{con}}$. 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 $\text{EN}$ to compute $\hat{E}(\tau_h(u))$.

**Finding PPND**

For a given seed set $S$, we use PPND$(u, S)$ to denote all PPND 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, $|\mathcal{P}_{PPND}(u, S)|$, grows exponentially with...
Finding $PP_{ND, C}(S)$ 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 threshold cannot be easily incorporated. To fill this gap, we develop an adapted Dijkstra (AD) algorithm for finding $PP_{ND, C}(S)$. Like Dijkstra Algorithm, AD adopts a greedy search strategy to select the shortest path for the destination. To satisfy the threshold 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 $PP_{ND, C}(S)$, $\text{Count}(u)$ recording the number of paths on node $u$, and $PH_{con}(u)$ recording $P_{con}$ of the found paths. 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). By extension. To satisfy the threshold constraint, AD only extends the path meeting the constraint in each iteration. Furthermore, we integrate the computation of path probabilities into AD.

Algorithm 2: Computing $\sigma(S)$ based on $PP_{ND}$

Input: $G=(V, E), T_{uw}, S, P_{uw}, f(\cdot), \theta$ and $C$

Output: $\sigma(S)$

1. Initialize $PP_{ND, C}(u, S) \leftarrow \emptyset, \text{Count}(u) \leftarrow 0$ and $PH_{con}(u) \leftarrow \emptyset$ for $u \in V$;
2. Initialize a min priority queue $PH$, and enqueue $P = (0, 1, path = \{u\})$ for $u \in S$;
3. while $PH \neq \emptyset$
   - $P \leftarrow$ dequeue $(T, P_{con}, path)$ from $PH$;
   - compute $E(\tau^P(u))$ according to $PH_{con}(u)$;
   - $P^P(u) \leftarrow P_{con} \times P_{uw} \times E(\tau^P(u))$;
   - if $\text{Count}(u) < C$ and $P^P(u) > \theta$ and $P$ is loopless then
     - $\text{Count}(u) \leftarrow \text{Count}(u) + 1$
     - $P_{con} \leftarrow P^P(u)$;
     - enqueue $P = (T + T_{uw}, P_{con}, path \cup \{v\})$ into $PH$ for $(u, v) \in E$;
4. for every $u$ with non-empty $PP_{ND, C}(u, S)$ do
   - $AP_{S}(u) \leftarrow 1 - \prod_{P \in PP_{ND, C}(u, S)} (1 - P^P(u))$
   - $\sigma(S) \leftarrow \sigma(S) + AP_{S}(u)$;
5. return $\sigma(S)$;

Experiments

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. The NetPHY is a collection network of papers, extracted from "Physics" sections in arXiv, and contains around 37,000 nodes and 181,000 edges.

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

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 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), 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. $P_{uv} = \frac{1}{\text{indegree}(v)}$, where indegree$(v)$ is indegree of node $v$. The expected influencing delay $T_{uv}$ of edge $uv$ follows the geometric delay distribution (Chen, Lu, and Zhang 2012). The parameter for geometric distribution is set at $\gamma (\text{indegree}(v) + 5)$. For simplicity, the maximum value of $T_{uv}$ is 15. If the generated de-
lay time $T_{uv} > 15$, $T_{uv}$ is reset as a random integer from 1 to 15. We also try other distributions for $T_{uv}$, such as poisson distribution and uniform distribution, and the results are similar. Empirically, threshold $\theta$ 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 $\gamma$.

**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 terminated MC if it ran 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 influence spread 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 achieves 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 although they achieve similar influence spread. 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 $\gamma$ on the NetHEPT and Wiki datasets. Figure 7 shows the influence spread for various $\gamma$ with $K = 50$. As expected, the influence spread grows as $\gamma$ increases. This is due to the fact that if $\gamma$ 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 $\gamma$. 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 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 based algorithm to efficiently compute influence spread of seed nodes. Performance of our algorithms is demonstrated with extensive experiments.

![Figure 5: The results of influence spread on four real world social networks.](image)

![Figure 6: The results of running time on four real world social networks.](image)

![Figure 7: The results of influence spread for different $\gamma$.](image)
References


