Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

DynaDiffuse: A Dynamic Diffusion Model for Continuous Time Constrained Influence Maximization

Miao Xie[1,2,3], Qiusong Yang[2,3], Qing Wang[2,3], Gao Cong[4], Gerard de Melo[5]
[1]University of Chinese Academy of Sciences, China, {xiemiao, qiusong, wq} @nfs.iscas.ac.cn
[2]National Engineering Research Center of Fundamental Software, Institute of Software, Chinese Academy of Sciences
[3]State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China.
[4]School of Computer Engineering, Nanyang Technological University, Singapore, gaocong@ntu.edu.sg
[5]Tsinghua University/Microsoft Research Asia, China, gdm@demelo.org

Abstract

Studying the spread of phenomena in social networks is critical but still not fully solved. Existing *influence maximization* models assume a static network, disregarding its evolution over time. We introduce the continuous time constrained influence maximization problem for dynamic diffusion networks, based on a novel diffusion model called DYNADIFFUSE. Although the problem is NP-hard, the influence spread functions are monotonic and submodular, enabling fast approximations on top of an innovative stochastic model checking approach. Experiments on real social network data show that our model finds higher quality solutions and our algorithm outperforms state-of-art alternatives.

1 Introduction

Understanding how information, diseases, etc. spread in a network is vital in many settings. Companies may wish to target users such that they, by word-of-mouth, influence their friends, friends-of-friends, and so on, to purchase a product. Finding a set of initial nodes that will eventually influence the largest number of nodes in a network is critical for achieving a high success rate (Domingos and Richardson 2001). Solving such an *influence maximization problem* can enable us to predict the success or failure of new innovations in their early stages, or to shape the process accordingly.

While several influence diffusion models have been proposed, all of them assume the given network is static (Guille et al. 2013). In reality, however, networks may evolve quite notably during the diffusion process (Mislove et al. 2008; Tang et al. 2010). Matsubara et al. (2012) found that the information diffusion process often goes on for days or even months. During this time, the degree of influence between two nodes can change quite substantially due to changing interests or social ties. Likewise, the network topology may change, with new nodes appearing or disappearing, and edges coming and going. Gomez-Rodriguez, Leskovec, and Schölkopf (2013) show that information diffusion networks inferred from cascade data often evolve dramatically as the information cascades accumulate. Accounting for such dynamics and thus pursuing seed node sets of a higher quality may lead to significant financial gains.

In this paper, we thus drop the assumption of a static network and investigate how to account for network evolution in influence maximization. We define the continuous time constrained influence maximization problem and propose the continuous time dynamic diffusion model (DYNADIFFUSE) for dynamic diffusion networks. Having captured specific dynamic characteristics of networks (such as herd behavior and activeness), DYNADIFFUSE enables us to consider their effects while estimating the influence spread of a set of nodes within a given time frame. This is achieved using an innovative stochastic model checking approach. Although the new problem is NP-hard, the influence spread function is monotonic and submodular. This enables us to obtain fast approximations with a lower bound precision ratio of $1 - 1/e$. In our experiments on real social network data, we show that network evolution indeed plays an important role in influence maximization. DYNADIFFUSE leads to significantly more influential node sets than previous models, with negligible time overhead.

2 Related Work

Kempe, Kleinberg, and Tardos (2003) first formulated the problem of influence maximization in a graph, in which influence propagation is captured using stochastic cascades, as in their independent cascade (IC) model. Recently, temporal aspects have been gaining increasing attention. Chen, Lu, and Zhang (2012) incorporated the time delay of influence diffusion into the IC model. Liu et al. (2012) defined a discrete time constrained information maximization problem and improved the IC model by adding the time stamps of every time period for each edge, but their algorithms do not consider continuous time constraints. Gomez-Rodriguez and Schölkopf (2012) proposed a continuous time constrained diffusion model, which can be inferred from a cascade database (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011). However, unlike DYNADIFFUSE, all aforementioned models assume a static network with fixed propagation rates or probabilities during the diffusion process.

Finding the best initial set of nodes requires combinatorial optimization, which is very costly. Many studies aimed at speeding up Kempe's greedy algorithm. Kimura and Saito (2006) introduced a shortest-path based IC model and provided efficient algorithms for it. Leskovec et al. (2007) proposed the *Cost-Effective Lazy Forward (*CELF*)* scheme

for selecting new seeds to significantly reduce the number of influence spread evaluations. Chen, Wang, and Yang (2009) proposed two algorithms: MixedGreedy and DegreeDiscount. Liu et al. (2014) developed a heuristic algorithm based on independent influence paths. Furthermore, Tang, Xiao, and Shi (2014) presented a greedy algorithm called TIM by generating many random graph sets under parameters control. Cohen et al. (2014) developed an efficient sketch-based algorithm (SKIM) by pre-generating a small sketch for nodes. However, these algorithms all assume static networks with fixed propagation probabilities.

3 Problem Definition and Overview

In contrast to previous work, our problem definition more closely reflects real networks by allowing diffusion rates and probabilities for each edge to vary during the diffusion.

Definition 1. Continuous time constrained influence maximization problem for dynamic diffusion networks: *Suppose we are given* (1) *a social network snapshot* $G(V, E)$, (2) *a set of dynamic characteristics reflecting how it will evolve,* (3) *a time bound* T, *and* (4) *a positive integer* $k < |V|$. *Our goal is to find an initial set* $I \subset V$ *of* k *nodes, maximizing the expected total number of nodes influenced up to time* T, *as estimated by an influence function* $\sigma_T(I)$.

Our solution to this problem consists of three parts:

1. In Section 4, we propose a new continuous time constrained dynamic diffusion model, called DYNADIFFUSE.
2. In Section 5.1, an influence spread analysis method for DYNADIFFUSE, relying on stochastic model checking, is given to estimate the influence of an initial set of nodes.
3. In Section 5.2, we design a fast greedy algorithm called FASTMARGIN after proving the monotonicity and submodularity of the influence function for DYNADIFFUSE.

4 Dynamic Diffusion Model

We now present a new dynamic diffusion model (DYNADIFFUSE) and then derive a specific instantiation based on real social network data.

4.1 Model Definition

In our diffusion model, each node is either influenced or not. Initially only a small set of nodes in an initial set I are influenced (by an external action such as supplying free product samples). Then, every node in I tries to influence its adjacent neighbors. If a node has successfully been influenced, it tries to influence its own neighbors, and so on. We assume that once a node has been influenced, this state persists until the end. The process terminates when there are no remaining uninfluenced nodes or when the given time T has elapsed.

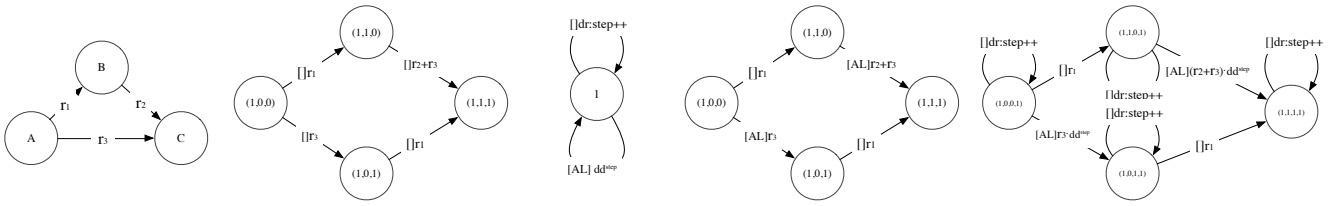
Whether a node succeeds in influencing its neighbor is modeled using propagation rates. Clearly, different edges have different propagation rates (Kossinets, Kleinberg, and Watts 2008), and these may also change over time. Following previous work, we use the exponential distribution to model the propagation rate. For an edge from node i to node j associated with a propagation rate r, the influence diffusion probability is $1 - e^{-rt}$, where t is the time span after node i is influenced. We can infer a diffusion network with potential diffusion edges and edge-specific propagation rates directly from diffusion cascade data using an optimization algorithm. Then we can construct a Continuous Time Markov Chain (CTMC) when given an initial set.

Definition 2. Continuous Time Markov Chain (CTMC): *A (labeled) CTMC is a tuple* (S, s_0, R, L), *where* S *is a finite set of states and* $s_0 \in S$ *is the initial state.* $R : S \times S \to R_{\geq 0}$ *is a transition rate matrix assigning rates to pairs of states with* $r \in R$ *used as rate parameters of the exponential distribution* $f(t) = re^{-rt}$,$t > 0$. *We allow self-loops. Finally, for a fixed finite set* P *of action labels,* $L : R \to 2^P$ *is a function assigning such labels to every transition with* $r > 0$.

We can create a CTMC (S, s_0, R, L), denoted by M_0, to represent the behavior of an inferred diffusion network $G(V, E)$ as follows. Each state $s \in S$ has $|V|$ dimensions, with values of 0 or 1 representing whether a node has been influenced. We first create an initial state s_0 for I, the initial set. For example, in Figure 1a and b, $s_0 = (1, 0, 0)$ represents that for $I = \{A\}$ only A is initially influenced. Next, we iteratively create all other reachable states in S and their corresponding transitions in R. Iterating over A's outgoing edges, we find that nodes B and C can be influenced, so we need two new states in S, $(1, 1, 0)$ for reaching B and $(1, 0, 1)$ for reaching C, along with two associated transitions for them. If these states are already in S, we just need to add transitions. We iterate this process until there is nothing left to add. For a transition from state s_1 to s_2, the transition rate is calculated as $\sum_{e_i} r_i$, where r_i is the propagation rate of each edge that can influence the new node. For example, A and B in state $(1, 1, 0)$ can independently influence C, so the transition rate is $r_2 + r_3$. Finally, we get four reachable states in S plus associated transitions in R, and $L = \emptyset$.

Next, we consider dynamic properties of social networks (Lang and Wu 2011, for example). It is impossible to model all possible dynamics exhibited by different social networks. However, having discovered salient dynamic characteristics from real data, we can capture these using additional CTMC models. We introduce local variables $f_1, \ldots, f_m$, which model dynamic factors for a dynamic characteristic. These factors are used to model the degree of effects on the diffusion network by a positive function $\Phi(f_1, \ldots, f_m)$. In the CTMCs for dynamic characteristics, transitions can be divided into two types: *Internal* transitions define how these dynamic factors evolve stochastically, while *external* transitions associated with action labels will be used for synchronization with the main CTMC M_0 defined earlier. M_0's action labels will be updated to match these labels.

For example, Figure 1c presents a dynamic characteristic called activeness (detailed in Section 4.3). There is only one state and one local variable step. The internal transition, $[]dr : \mathrm{step}{+}{+}$, represents that step increases with rate dr, where dr is a constant. The external transition, $[AL]dd^{\mathrm{step}}$, is labeled with an action label AL indicating low activeness. After defining dynamic characteristics, we update the corresponding action labels for M_0. If node C has low activeness in Figure 1a, we add the action label AL for all transitions that can influence C. The result is shown in Figure 1d. We next explain how to connect these CTMCs to model the evo-



(a) Diffusion Network G (b) CTMC for G, initial set I={A}, (1,0,0) is initial state (c) CTMC for Activeness (d) Correlate with (c) by adding action labels for (b) (e) Parallel Composition for (c) and (d)

Figure 1: DYNADIFFUSE modeling example, where dd stands for decreaseDelta and dr for decreaseRate

lution of the inferred network.

Definition 3. CTMC Parallel Composition: *For two labeled CTMCs $M_1 = (S_1, s_{01}, R_1, L_1)$ and $M_2 = (S_2, s_{02}, R_2, L_2)$, the parallel composition, $M_1 \parallel M_2$, is a CTMC $(S_1 \times S_2, (s_{01}, s_{02}), R_c, L_1 \cup L_2)$, where R_c is defined as:*

$$\frac{[\iota]s_1 \xrightarrow{\alpha}_1 s_1'}{[\iota]\langle s_1, s_2\rangle \xrightarrow{\alpha} \langle s_1', s_2\rangle} \quad \iota \in L_1 \setminus L_2, \alpha \in R_1$$

$$\frac{[\iota]s_2 \xrightarrow{\alpha}_2 s_2'}{[\iota]\langle s_1, s_2\rangle \xrightarrow{\alpha} \langle s_1, s_2'\rangle} \quad \iota \in L_2 \setminus L_1, \alpha \in R_2$$

$$\frac{[\iota]s_1 \xrightarrow{\alpha_1}_1 s_1' \wedge [\iota]s_2 \xrightarrow{\alpha_2}_2 s_2'}{[\iota]\langle s_1, s_2\rangle \xrightarrow{\alpha_1 \cdot \alpha_2} \langle s_1', s_2'\rangle} \quad \iota \in L_1 \cap L_2, \alpha_1 \in R_1, \alpha_2 \in R_2$$

Here, $[\iota]s \xrightarrow{\alpha} s'$ means that the CTMC evolves from state s to s' when $\alpha \in R$ is performed with action label $\iota \in L$.

We can see that transitions with different action labels in different M_i may only occur independently. In contrast, transitions with the same action labels in different M_i must occur together with a combined transition rate $\prod_i \alpha_i$. The combined CTMC with the activeness dynamic characteristic is shown in Figure 1d. In this example, if step is changed during the simulation, the transition rates of subsequent transitions will evolve as $[AL]r_3 \cdot dd^{\text{step}}$.

In this way, we can model most common dynamic characteristics. If a new connection appears, we can introduce an edge with a minimal positive value at the beginning that increases later. An edge (re-)emerges when its rate increases from the minimal value, and conversely an edge disappears if its rate decreases to that value. Nodes can be regarded as removed when all of their edges have disappeared.

With this modeling approach, we get a labeled CTMC M_0 for an inferred diffusion network and a series of CTMCs $D = \{d_1 ... d_n\}$ for dynamic characteristics. Finally, we define the DYNADIFFUSE model as follows.

Definition 4. Continuous Time Dynamic diffusion Model (DynaDiffuse): *Given a CTMC M_0 for a network $G = (V, E)$, a set A of propagation rates $\alpha_{u,v}$ for each $(u, v) \in E$, an initial set of vertices $I \subseteq V$, and a set of dynamic characteristics $D = \{d_1, \ldots, d_n\}$, where each $d_i \in D$ is a CTMC model whose transition rate matrix represents a given positive function $\Phi(f_1, \ldots, f_m)$ (with dynamic factors f_j and $\Phi \geq 0$),* DYNADIFFUSE *M is a synchronous CTMC model defined as $M = M_0 \parallel D = M_0 \parallel d_1 \parallel \cdots \parallel d_n$.*

4.2 Modeling Process

Algorithm 1 presents the complete modeling process for a DYNADIFFUSE instantiation. First, we make use of a convex optimization algorithm (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011) to infer the diffusion network topology $G = (V, E)$ with $\bar{R}$ as a set of propagation rates for every edge (Figure 1a). Given an initial set, we can construct its corresponding CTMC $M_0 = (S, I, R, L)$ where L is empty, i.e., $L(r) = \emptyset$ for all $r \in R$. Then, we model each dynamic characteristic by a dynamic module $d_i \in D$ as a CTMC (Figure 1c). To establish their correlation, we add action labels L of M_0 for corresponding external transitions in d_i (Figure 1b). Finally we synchronize them by following Definition 3 (Figure 1e). We can then use stochastic model checking to predict the influence spread in a given time (Section 5.1).

Algorithm 1 Complete Modeling Process

Input: C (Cascade Data), T (time constraint), I (initial set)
1: $G(V, E, \bar{R}) \leftarrow gls(C, T)$ # optimization algorithm
2: Create CTMC $M_0 = (S, I, R, L)$ from G, A
3: $D \leftarrow$ dynamic modules mined from real data
4: modify L of M_0 to capture the correlation with D
5: **return** $M_0 \parallel D$ as DynaDiffuse

4.3 Salient Dynamic Characteristics

While DYNADIFFUSE can flexibly capture various kinds of network evolution, we focus on two important characteristics of information spread in social networks.

Herd Behavior. It is a well-established phenomenon that over time, with an increasing number of influenced people, propagation rates increase. The strength of this effect, however, differs for different types of users. Several studies show that information diffusion is affected by a person's conformity, i.e., inclination to be influenced (Li, Bhowmick, and Sun 2011; Tang, Wu, and Sun 2013; Raafat, Chater, and Frith 2009). For each user v, we compute a conformity score

$$\frac{|\{(m, v, t) \in P_v \mid \exists (m, v', t') : p_{v,v'} \wedge \epsilon \geq t - t' \geq 0\}|}{|P_v|}$$

where P_v is the set of user v's posting history logs, $p_{v,v'} \in P$, representing the posting logs that user v posted to user v' and ϵ is a maximal time difference for v's reposting of

v''s posting (one week in our dataset from Section 6). Individual conformity represents how strongly v's posting behavior conforms to that of her friends. In our dataset, we discover that nearly 30% of users have a conformity > 0.6. We divided the users into three groups according to their conformity values and randomly selected 100 users from each group to calculate the distribution of the times a posting had been reposted before they forwarded it. Our analysis revealed that users with high conformity either repost very popular postings or brand new ones posted by their close friends, often related to their personal offline life. Low conformity users, in contrast, do not seem to show any clear pattern. Therefore, we model herd behavior at different intensities for different conformity levels. Given a conformity vector for all nodes, we define a CTMC as follows.

module "herd behavior"
$n : [0..|V|]$ init by $|I|$;
/* conformity $\geq 60\%$ */
$[CH]\ n > 0 \rightarrow 1 + \psi(n) * \text{conformity}_{\text{high}} : (n' = n + 1)$;
/* $60\% >$ conformity $\geq 30\%$ */
$[CM]\ n > 0 \rightarrow 1 + \psi(n) * \text{conformity}_{\text{middle}} : (n' = n + 1)$;
/* conformity $< 30\%$ */
$[CL]\ n > 0 \rightarrow 1 + \psi(n) * \text{conformity}_{\text{low}} : (n' = n + 1)$;
end module

The module definition syntax used here is based on the Reactive Modules formalism (Alur and Henzinger 1999), with local variables and transition commands that reflect the operational behavior of the module, i.e., state transitions of the CTMC. Here, the local variable n (with range from 0 to $|V|$) represents the current number of influenced nodes, which is initialized as $|I|$. The form of each transition command is $[\text{action label}]\ \text{guard condition} \rightarrow \text{rate} : \text{updates}$. Updates can be performed if and only if the current state of the CTMC meets the guard condition. We label each node's incoming edge by its conformity level in the CTMC for the inferred diffusion network, with $[CH]$ for high, $[CM]$ for medium, and $[CL]$ for low conformity. For each level i, conformity_i is the average value for all nodes at this level. The transition rate function $\Phi(n) = 1 + \psi(n) \cdot \text{conformity}_i$ models the changing trend of the rate as the number of influenced users increases. We define $\psi(n) = \gamma \left(1 + e^{-\delta\left(\frac{n}{|V|} - \text{b}\right)}\right)^{-1}$ for $\gamma > 0$, as this mirrors the spread process for popular events in a network, first increasing gradually at an increasing rate and then stabilizing. The range of $\psi(n)$ is determined by γ, the curve's slope by δ, and b is the rate's breakpoint. In our experiments, we study how these variables affect the initial set selection for influence maximization.

Activeness of Actors. Studies (Lang and Wu 2011; Viswanath et al. 2009) show that social actors exhibit different levels of activeness. In particular, highly active ones contribute more towards the interaction, e.g. by reposting messages and posting new stories. Although there are many other factors, such as user interests, beliefs, and topics, much of their effects can be attributed to a user's activeness.

From our real diffusion dataset (cf. Section 6), we first measure the activeness of every node $v \in V$ as $\text{activeness}(v) = |v_p|$, i.e., the total number of messages

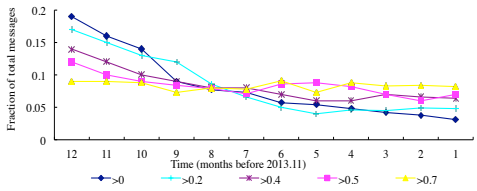


Figure 2: Fraction of interactions over time for one year .

the user has posted or reposted (during a one week period in our experiments). For analysis purposes, we divided the users into groups according to their activeness values and randomly selected 10% of users in each group to calculate the average interaction levels based on one year of historic data. Figure 2 illustrates that highly active users display a relatively more even distribution of posting messages over time, compared to other groups. On the contrary, less active users post less and less over time. The trends are different for different groups. To capture this, we thus divide the user base into two parts using a threshold on their activeness. For active nodes (whose activeness is larger than the threshold), we assume stable transition rates for all of their incoming edges during the diffusion process. On the contrary, the incoming edges' transition rates for inactive nodes (low level activeness) will stochastically decrease over time, following an exponential distribution with expectation decreaseRate. Given a classification threshold γ, the function Φ of this dynamic module can be expressed as

$$\Phi = \begin{cases} \text{decreaseDelta}^{\text{step}} & \text{activeness} < \gamma \\ 1 & \text{otherwise} \end{cases}$$

Then, given an activeness vector of all nodes, we define the activeness module using a label AL for low activeness and a parameter decreaseRate to control the evolution.

module "Activeness"
const float decreaseRate;
const float decreaseDelta;
step : [0..IntMax] init by 0;
[] step $\geq 0 \rightarrow$ decreaseRate : (step$'$ = step + 1);
$[AL]$ step $> 0 \rightarrow \text{decreaseDelta}^{\text{step}} : (\text{step}' = \text{step})$;
end module

5 Influence Analysis and Maximization

We now define the influence function for the DYNADIFFUSE model and apply our novel stochastic model checking approach to predict the influence spread.

5.1 Spread Analysis

Definition 5. *The **influence spread** $\sigma_T(I)$ of an initial set I until time T in a* DYNADIFFUSE *model M_I is defined as the expected number of nodes influenced until T:*

$$\sigma_T(I) = \mathbb{E}N_T(I) = \sum_{i=1}^{|V|} P(t_i \leq T | M_I),$$

where t_i denotes the time when node i is influenced.

To estimate the influence spread of a given initial set I, we rely on stochastic model checking (Kwiatkowska, Norman, and Parker 2007), a method for computing possible behaviors of a stochastic system. Given a model description and a formula in temporal logic, stochastic model checking provides the likelihood of the model satisfying the formula as well as a wider range of quantitative measures relating to model behavior. For example, one can use stochastic model checking to determine the expected running time or expected number of lost messages of a system.

Transitions (except self-loops) of the DYNADIFFUSE model correspond to non-influenced nodes becoming influenced. To calculate $\mathbb{E}N_T(I)$, we tie a reward to every such transition, and accumulate all rewards in the diffusion process to estimate the number of influenced nodes. Given a DYNADIFFUSE model M, we denote the instantaneous reward associated with the transition from state i to state j as reward_{ij}. The reward Markov process associated with M is then $(M, \mathbb{RE}_M)$, where the reward accumulated in the time interval $(0,t]$ is $\mathbb{RE}_M = \int_0^t \text{reward}_{M(u-),M(u)} dN_M(t)$, where $N_M(t)$ is the number of state transitions of M in the time interval $(0,t]$. Assume $\text{reward}_{ij} = 1$ for all transitions. Then $\mathbb{RE}_M = \int_0^t dN_M(t) = \sum_{i=1}^{|V|} P(t_i \leq T|M_I)$. This can be determined using a Uniformisation based Stochastic Model Checking approach (Kwiatkowska, Norman, and Parker 2007). It allows us to either get an exact result by verification, or to estimate the result using a faster simulation approach. In our study, we used the PRISM system (Kwiatkowska, Norman, and Parker 2011) for this.

5.2 Scalable Influence Maximization

Complexity, Monotonicity, and Submodularity.

Theorem 1. *Continuous time constrained influence maximization on a* DYNADIFFUSE *model is* NP-hard.

Proof. With dynamic modules $D = \emptyset$ and $T \to \infty$, our DYNADIFFUSE model turns out to be an instance of the independent cascade model. Hence, the latter is a special case of DYNADIFFUSE. As the traditional influence maximization problem is known to be NP-hard (Kempe, Kleinberg, and Tardos 2003), this new problem is also NP-hard. □

Theorem 2. *The influence function $\sigma_T(I)$ for* DYNADIFFUSE *is monotonous.*

Proof. If initial nodes are influenced at $t = 0$, each transition (except self-loops) of a DYNADIFFUSE model means there will be a new influenced node. The time delay is determined by an exponential distribution with a step-varied expectation value. Consider the possible distributions of all possible time differences between each pair of states in the model. There will be a series of different state graphs χ. For a given $\Delta x \in \chi$, we define $\sigma_{T,\Delta x}(I)$ as equal to the number of nodes reachable from I by at least one path with length no larger than T. It is very natural to discover that $\sigma_{T,\Delta x}(S) \leq \sigma_{T,\Delta x}(S \cup v)$ (for $S \subset V$). Therefore, $\sigma_{T,\Delta x}(I)$ is monotonous. As $\sigma_T(I) = (\sum_{x\in\chi} \sigma_{T,\Delta x}(I))/|\chi|$, we obtain that $\sigma_T(I)$ is monotonous as well. □

Theorem 3. *The influence function $\sigma_T(I)$ for* DYNADIFFUSE *is submodular.*

Proof. Consider any set of nodes $V_1 \subseteq V_2 \subseteq V$, and a node $v_0 \in V \setminus V_2$ (if $V_2 = V$, then trivially $V_{01,T} = V_{02,T}$ below). We again use $\Delta x \in \chi$, as in the proof of Theorem 2. We assume $\Delta V_{0i,T}$ is the set of nodes that can be reached from v_0, but cannot be reached by V_i $(i = 1, 2)$ within the T bound in the Δx graph. Since $V_1 \subseteq V_2$, we obtain $\Delta V_{01,T} \geq \Delta V_{02,T}$. We see that $\sigma_{T,\Delta x}(V_1 \cup \{v_0\}) - \sigma_{T,\Delta x}(V_1) = \Delta V_{01,T} \geq \Delta V_{02,T} = \sigma_{T,\Delta x}(V_2 \cup \{v_0\}) - \sigma_{T,\Delta x}(V_2)$ and thus $\sigma_{T,\Delta x}(I)$ is submodular. From this it follows that $\sigma_T(I)$ is submodular, concluding the proof. □

FASTMARGIN Greedy Algorithm. The standard greedy algorithm for maximizing monotonic and submodular functions has a provable lower bound ratio of $1 - 1/e$. However, for genuine scalability, we propose a faster marginal influence spread estimation algorithm for the continuous time diffusion model. Our algorithm (Algorithm 2) first predicts a relatively precise influence spread using stochastic model checking on the CTMC Parallel Composition. Then, we predict the increased marginal influence spread $(\sigma_T(I \cup \{v\}) - \sigma_T(I))$ of adding each $v \in V \setminus I$ when selecting an additional node to be added to the current initial set. To improve the efficiency, we estimate this marginal influence spread using a faster discounted formula for $\sigma_T(\{v\})$ in Line 12, instead of determining a precise value with stochastic model checking.

Algorithm 2 FASTMARGIN Greedy algorithm

Require: a DYNADIFFUSE for a graph $G = (V, E)$, the size k of the desired initial set, a time constraint T

1: $I \leftarrow \emptyset$;
2: **for** each $v \in V$ **do**
3: predict accurate $\sigma_T(\{v\})$ using stochastic model checking
4: $u \leftarrow \text{argmax}_v(\sigma_T(\{v\}))$
5: $I \leftarrow \{u\}$
6: **for** $i \leftarrow 2$ to k **do**
7: **for** each $v \in V \setminus I$ **do**
8: **if** $(v, u) \in E$ **and** $u \in N(I)$ **then**
9: $P_{I,u} \leftarrow 1 - \Pi_{(w,u)\in E, w\in I}(1 - e^{-r_{u,w}(T)})$
10: **else**
11: $P_{I,u} \leftarrow 0$
12: $\Delta\sigma_T(\{v\}) \leftarrow \sigma_T(I \cup \{v\}) - \sigma_T(I)$
$\approx \sigma_T(\{v\}) \frac{\Sigma_{(v,u)\in E}(1-e^{-r_{v,u}(T)})(1-P_{I,u})\sigma_T(\{u\})}{\Sigma_{(v,u)\in E}(1-e^{-r_{v,u}(T)})\sigma_T(\{u\})}$
13: $u \leftarrow \text{argmax}_v(\Delta\sigma_T(\{v\}))$ # using CELF
14: $I \leftarrow I \cup \{u\}$
15: **return** I # best set I

Here, $N(I)$ is the set of directed successors of I, and $P_{I,u}$ is the probability that u is influenced immediately by current initial nodes. We estimate the probability with an exponential distribution for the whole diffusion period. The discount is to subtract the influence spread that may be obtained by I from $\sigma_T(\{v\})$. The higher the probability that v's neighbors are already influenced by I, the larger the discount that should be applied to $\sigma_T(\{v\})$. For scalability to large networks, we also adopt the CELF lazy evaluation approach

(Leskovec et al. 2007), which dramatically reduces the number of evaluations of $\text{argmax}_v(\sigma_T(I \cup \{v\}) - \sigma_T(I))$. We call this algorithm FASTMARGIN. This algorithm can flexibly be used in any continuous-time constrained diffusion model that assumes that the diffusion rate follows the exponential distribution.

Time and space complexities. Let $d_{\max}$ be the maximum degree of all nodes, and S and R be the states and transition matrix of a DYNADIFFUSE instance, respectively. In Algorithm 2, the first for loop requires $O(|V||S|^2)$ time for uniformisation. The second for loop will cost $O((k-1)|V|d_{\max})$. So the total runtime complexity is $O(|V||S|^2 + (k-1)|V|d_{max})$. However, the standard greedy algorithm's time complexity is $O(k|V||S|^2)$. Thus, they have the same running time when $k = 1$, and Algorithm 2 is faster for $k > 1$. The space complexity is $O(|V|(|S| + |R|))$, dominated by stochastic model checking in line 3.

6 Experimental Evaluation

Dataset. We evaluate our model and algorithm on a microblog diffusion dataset from Sina Weibo, a major Chinese microblogging platform. The data covers $96,700$ users and $373,412$ microblog postings ($211,052$ original and $162,360$ reposted) from the time period of November 2–8, 2013. For each of these postings, it contains a set of time-stamped reposting logs with user ID, reposting time, and reposting contents, in total nearly $4,000,000$ entries. From this data, we inferred an underlying diffusion network with $6,000$ top nodes and the $30,000$ fastest edges using the algorithm from Gomez-Rodriguez, Balduzzi, and Schölkopf (2011), and calculated the conformity and activeness vectors.

Comparing Models. Figure 3a shows that the seeds' influence spread obtained by DYNADIFFUSE (i.e., FastMargin-Dynamic, CELF-Dynamic) is consistently superior to that of the *static* continuous time diffusion model (Gomez-Rodriguez and Schölkopf 2012) (i.e., FastMargin–Static, CELF–Static), as well as the IC model (Kempe, Kleinberg, and Tardos 2003) (i.e., CELF-IC). The FASTMARGIN algorithm is slightly less precise than the pure CELF algorithm because of its heuristics, but the gap is not large. We additionally compare the DYNADIFFUSE-estimated influence spread with the *real influence spread* in the social network. To assess the real diffusion, we first crawled all original microblogs published on November 9, 2013 for every user in the obtained initial set and then calculated the average total number of repostings in the following week ($T = 1$ to 7). We regard the average total number of repostings as the influence spread for each user. We only calculate the reposting numbers within our data set (96,700 users) and minus the overlap. Figure 3b illustrates that the estimated results of our dynamic model, i.e., FastMargin–Dynamic (Slow, Quick), are closer to the real situation than those of the static diffusion model, FastMargin–Static. Moreover, the evolution speed parameters in our model influence the precision of the estimated results. To get even more precise results, we could easily tune them with collected historic data. Note that DYNADIFFUSE consistently outperforms existing models across different values of these parameters.

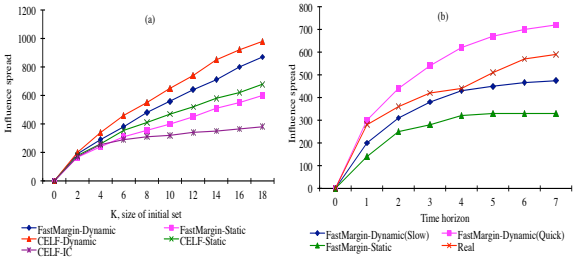


Figure 3: (a) Diffusion models comparison by DYNADIFFUSE with $T = 3$ days. Herd behavior module: $\delta = 30$, $b = 0.05$, activeness module: decreaseDelta $= 0.9$, decreaseRate $= 0.2$. (b) Comparison by the real influence spread with $K = 6$. Slow evolution: $\delta = 30$, $b = 0.05$, decreaseDelta $= 0.9$, decreaseRate $= 0.2$. Quick evolution: $\delta = 100$, $b = 0.01$, decreaseDelta $= 0.9$, decreaseRate $= 0.4$. Legend: greedy algorithm–diffusion model

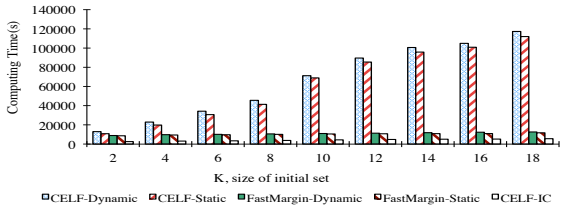


Figure 4: Running time of algorithms ($T = 6$ days)

Comparing Algorithms. We then evaluated the efficiency and scalability of DYNADIFFUSE and the FASTMARGIN algorithm. Figure 4 shows running time on a workstation (2.5GHz Dual core, 4GB RAM). FastMargin–Dynamic and FastMargin–Static are significantly faster than CELF (CELF–Dynamic and CELF–Static) and are more scalable as K grows. Interestingly, the time gap between DYNADIFFUSE (i.e., CELF–Dynamic, FastMargin–Dynamic) and the static model (i.e., CELF–Static, FastMargin–Static) is very small. The analysis for DYNADIFFUSE with FASTMARGIN is thus just slightly more costly than for the IC model, yet produces an initial set of significantly higher quality. Note that FASTMARGIN's running time remains almost constant as k increases. CELF-IC has similar behavior, but with worse output quality, as we have seen earlier.

7 Conclusion and Future Work

We have presented the novel continuous constrained influence maximization problem for dynamic networks, and introduced a novel diffusion model called DYNADIFFUSE for it. Under DYNADIFFUSE, we employ an innovative stochastic model checking approach to considering dynamic effects and estimating the influence spread of a set of nodes within a given time frame. Our experiments show that network dynamics indeed have strong effects on the solution's quality. In the future, we will investigate how to learn dynamic behavior from data. Additionally, we believe that our novel approach of relying on stochastic model checking can serve as an important new framework for many new forms of social network analysis.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grants 91318301, 91218302, 61432001, 61303163.

References

Alur, R., and Henzinger, T. A. 1999. Reactive modules. *Formal Methods in System Design* 15(1):7–48.

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

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

Cohen, E.; Delling, D.; Pajor, T.; and Werneck, R. F. 2014. Sketch-based influence maximization and computation: Scaling up with guarantees. In *Proc. CIKM*, 629–638.

Domingos, P., and Richardson, M. 2001. Mining the network value of customers. In *Proc. SIGKDD*, 57–66.

Gomez-Rodriguez, M., and Schölkopf, B. 2012. Influence maximization in continuous time diffusion networks. In *Proc. ICML*, 313–320.

Gomez-Rodriguez, M.; Balduzzi, D.; and Schölkopf, B. 2011. Uncovering the temporal dynamics of diffusion networks. In *Proc. ICML*, 561–568.

Gomez-Rodriguez, M.; Leskovec, J.; and Schölkopf, B. 2013. Structure and dynamics of information pathways in online media. In *Proc. WSDM*, 23–32.

Guille, A.; Hacid, H.; Favre, C.; and Zighed, D. A. 2013. Information diffusion in online social networks: a survey. *SIGMOD Rec.* 42(1):17–28.

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

Kimura, M., and Saito, K. 2006. Tractable models for information diffusion in social networks. In *Proc. PKDD*, 259–271.

Kossinets, G.; Kleinberg, J.; and Watts, D. 2008. The structure of information pathways in a social communication network. In *Proc. SIGKDD*, 435–443.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2007. Stochastic model checking. In *Proc. SFM*, 220–270.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. CAV*, 585–591.

Lang, J., and Wu, S. 2011. Social network user lifetime. In *Proc. ASNAM*, 289–296.

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

Li, H.; Bhowmick, S. S.; and Sun, A. 2011. Casino: towards conformity-aware social influence analysis in online social networks. In *Proc. CIKM*, 1007–1012.

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

Liu, B.; Cong, G.; Zeng, Y.; Xu, D.; and Chee, Y. M. 2014. Influence spreading path and its application to the time constrained social influence maximization problem and beyond. *TKDE* 26(8):1904–1917.

Matsubara, Y.; Sakurai, Y.; Prakash, B. A.; Li, L.; and Faloutsos, C. 2012. Rise and fall patterns of information diffusion: model and implications. In *Proc. SIGKDD*, 6–14.

Mislove, A.; Koppula, H. S.; Gummadi, K. P.; Druschel, P.; and Bhattacharjee, B. 2008. Growth of the flickr social network. In *Proc. WOSN*, 25–30.

Raafat, R. M.; Chater, N.; and Frith, C. 2009. Herding in humans. *Trends in cognitive sciences* 13(10):420–428.

Tang, J.; Musolesi, M.; Mascolo, C.; and Latora, V. 2010. Characterising temporal distance and reachability in mobile and online social networks. *SIGCOMM Comput. Commun. Rev.* 40(1):118–124.

Tang, J.; Wu, S.; and Sun, J. 2013. Confluence: Conformity influence in large social networks. In *Proc. SIGKDD*, 347–355.

Tang, Y.; Xiao, X.; and Shi, Y. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proc. SIGMOD*, 75–86.

Viswanath, B.; Mislove, A.; Cha, M.; and Gummadi, K. P. 2009. On the evolution of user interaction in facebook. In *Proc. WOSN*, 37–42.