Influence Diffusion in Online Social Networks With Propagation Rate Changes

Tianyi Pan, Xiang Li, *Member, IEEE*, Alan Kuhnle, *Member, IEEE*, and My T. Thai, *Member, IEEE*

***Abstract*—Information can propagate among Online Social Network (OSN) users at a high speed, which makes the OSNs important platforms for viral marketing. Although the viral marketing related problems in OSNs have been extensively studied in the past decade, the existing works all assume known propagation rates. In this paper, we propose a novel model, Dynamic Influence Propagation (DIP), which allows propagation rates to increase after a topic becomes popular and can be used for describing information propagation in OSNs more realistically. Based on DIP, we define a new research problem: Threshold Activation Problem under DIP (TAP-DIP). However, it adds another layer of complexity over the already #P-hard TAP problem. Despite it hardness, we are able to approximate TAP-DIP with $O(\log |V|)$ ratio, where $|V|$ is the number of users in the network. Our solution consists of global optimization techniques and a novel solution to the general version of TAP. We also consider the more complicated case when the propagation rates may change multiple times and the changes are non-immediate, with corresponding solution and analyses. We test our solution using various real OSN datasets, and demonstrate that our solution not only generates high-quality seed sets, but also scales.**

***Index Terms*—Dynamic influence propagation, online social network, threshold activation problem.**

I. Introduction

INFLUENCE propagation is an essential problem that has been studied in various contexts [1]–[28]. In OSNs, popular and unpopular topics may propagate following completely different patterns. Especially, people tend to share the popular topics with their friends much faster. By analyzing the retweet delay for tweets before/after a topic becomes trending, we observed a much shorter retweet delay after trending for the majority of around 4,000 Twitter trending topics in the US. This observation indicates that influence propagation speed can depend on the topic's popularity.

This characteristic of influence propagation has not been studied in literature. In previous works, there are two major types of influence propagation models: 1) the Triggering Model [3] in which the influence propagates in rounds and thus the propagation rate is uniform; 2) the Continuous-Time Diffusion Model [7], in which the propagation rate is decided by a probability density function (pdf) for each edge. The two existing models share a common feature that they are *static:* whether the propagation rate is constant or follows a pdf, it is known before the propagation starts.

To better depict influence propagation in reality, it is necessary to develop a propagation model that enables changes of propagation rates *based on the current propagation status*. Therefore, we propose the Dynamic Influence Propagation (DIP) model, which can explicitly consider the rate changes. In the DIP model, we extend the idea in literature [18] that a topic becomes more popular when the number of influenced nodes increases. Notice that the condition may not always reflect the complicated conditions in reality (e.g. Twitter has internal algorithms for selecting trending topics), but it is a reasonable abstraction. We then formulate the Threshold Activation Problem with DIP (TAP-DIP) to analytically study the model. TAP-DIP asks for a seed set with a minimum size that guarantees the number of nodes being influenced can reach a threshold within time limit. In the problem, the propagation rate may change due to the DIP model.

The main challenge of TAP-DIP is resulted from its new dynamic propagation rates, as it creates an obstacle for using the sampling techniques [3], [9] that are applied to solve influence propagation related problems. The sampling techniques are important for influence propagation since even computing the exact influence is #P-hard [5]. Each sample provides information on what nodes can be influenced by a certain node (forward sampling [3]) or the set of nodes that may influence a target node (reverse sampling [9]). As the sampling techniques only work with a fixed propagation model, they have no access to dynamic information such as propagation rate change and cannot be easily adapted to solve TAP-DIP.

To tackle the challenges brought by dynamic propagation rates, for a special case that the propagation rate may change once, we propose the algorithm FAST (stands for Finding Anticipated Speedup Time) which can decide the near-optimal time that the propagation rate may increase, with a carefully designed global optimization framework utilizing Lipschitz-alike

Manuscript received February 18, 2020; revised July 4, 2020; accepted August 7, 2020. Date of publication August 11, 2020; date of current version December 30, 2020. This work was supported in part by DTRA HDTRA1-14-1-0055, NSF CNS-1814614 and NSF CNS-1948550. Recommended for acceptance by Dr. Yingshu Li. *(Corresponding author: My T. Thai.)*

Tianyi Pan is with the Google Inc, Mountain View, CA 94043 USA (e-mail: tianyipan@google.com).

Xiang Li is with the Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA 95053 USA (e-mail: xli8@scu.edu).

Alan Kuhnle is with the Department of Computer Science, Florida State University, Tallahassee, FL 32306 USA (e-mail: akuhnle@fsu.edu).

My T. Thai is with the Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32612 USA (e-mail: mythai@cise.ufl.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TNSE.2020.3015935

properties. The optimality is in terms of minimizing the number of seeds used for two objectives: 1) trigger the propagation rate increase 2) ensure total number of influenced users reach the threshold. FAST breaks down TAP-DIP into subproblems in which the rate increases happen at fixed times, and hence the sampling methods are again applicable. However, the subproblems are still complicated as they need to meet the thresholds for both triggering the rate increase and satisfying the activation requirement. To solve the subproblems, we designed the first efficient algorithms for both Multi-TAP (MTAP) and Multi-Influence Maximization (MIM). FAST can solve the TAP-DIP problem with approximation ratio $2\log|V|$ (V is the set of nodes in the OSN), which is close to the best ratio $\log|V|$ that one can expect for TAP *without* DIP.

We also consider the general setting with multiple changes of propagation rates. To solve it, we propose an efficient multi-variate global optimization algorithm and utilize the proposed algorithms for MTAP and MIM. The approximation ratio of this solution is $(1+\zeta)\log|v|$, where ζ is the number of times that the propagation rate may change.

In summary, our contributions are as follows.

- We propose TAP-DIP, the first influence propagation related problem in OSNs that explicitly considers propagation rate increases. We support the validity of the model by data analysis results from crawled retweets in around 4,000 Twitter trending topics.
- We propose the algorithm FAST to solve TAP-DIP in the special case with one possible rate change. It is the first solution to TAP-DIP with an approximation ratio of $2\log|V|$. The two subroutines of FAST, MMinSeed and Multi-IM, are the first algorithms that can efficiently solve the MTAP problem and the MIM problem, respectively.
- We propose the FASTS algorithm to solve the general problem that allows ζ propagation rate changes in TAP-DIP, with the approximation ratio $(1+\zeta)\log|V|$.
- We perform extensive experiments on various real OSN data sets to demonstrate both the efficiency of our proposed algorithms and the drastic difference in the solutions when considering rate increase.

Related Work. Kempe *et al.* [3] are the first to study influence propagation in OSNs mathematically. Their focus was on the Influence Maximization problem (IM), which drew much attention in the research community [4]–[14], [22], [27]. Recently, IM related studies also consider solution robustness [24], competition with time constraint [25], IM with empirical models [26], in multiple rounds [23] and at community level [28]. Another major problem is TAP [15]–[20]. The main propagation model adopted in the papers is the Triggering model [3] or its variations, the Independent Cascading (IC) model or the Linear Threshold (LT) model. Another model that considers variation in propagation rate is the continuous time diffusion model [7]. However, both models assume known and fixed parameters for the diffusion, which may not represent the real-world scenarios.

As even computing the exact influence is #P hard [5], the mainstream approach of solving IM or TAP relies extensively on sampling, which is inefficient (due to many redundant samples) until Borgs *et al.* proposed the Reverse Influence Sampling (RIS) method in [9]. The RIS method was further refined in [10]–[12], [14] for better time complexity. However, the RIS method was not yet applied to solve TAP or MIM, nor can it consider the DIP model. The only exception is [21], from which this paper is extended.

Organization. The rest of the paper is organized as follows. In Section II, we present our analysis on propagation rates and define the TAP-DIP problem. Section III and IV discuss our solution, FAST to TAP-DIP. Section V discusses the solution FASTS to the generalized problem. The performance of FAST/FASTS and the behavior of the DIP model are analyzed in Section VI. Section VII concludes the paper.

II. Model And Problem Definition

A. Analysis of Twitter Data

We crawled the tweet stream data for 5,049 different Twitter trending topics in the US using the REST APIs[1], during the period of Nov. 2016 to Apr. 2017. Specifically, we collected the *retweets* whose times are within three days of the time that the topic *first* became trending. In order to decide the trending times for the topics, we first maintain the collection of all trending topics in three days and then crawl the current trending topics every 5 minutes. The trending time is considered as the first time that a new trending topic is recorded. We also update the collection when necessary. When the trending time of a topic is decided, we can use the Search API in REST[2] to fetch the historical retweets within the desired times. The retweets are separated by the trending time and into two groups, before trending and after trending, as our major goal is to demonstrate that the propagation rate increases after the topic being trending. The propagation rate is characterized by the reciprocal of the time difference between the retweet and the original tweet (retweet delay) in this case. We omit the topics having less than 100 retweets before/after trending to avoid outliers (such as promoted trending topics with few retweets) and we are left with 3,988 topics after this step. For each remaining topic, we calculated the time difference between each of its retweets and the corresponding original tweet. Based on the arrays of time differences before/after trending, we can decide whether the time difference decreased (or equivalently, propagation rate increased) after trending, using KS-test [29] and t-test.

In Fig. 1, we present the test results of all the topics. It is clear that the increase of propagation rate after trending is a common phenomenon, as 86.1% of the topics have shorter average retweeting time delay. Among those topics (label "L"), 72.8% have significant increase in propagation rate ("L, NKS,SL"), verified by both KS-test and t-test.

We further study the distribution of propagation rate increase of those who increased significantly (2,502 topics in total). Define the propagation rate increase as

[1] https://dev.twitter.com/rest/public
[2] https://dev.twitter.com/rest/public/search

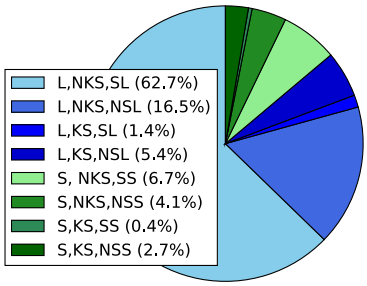


Fig. 1. Statistical Test Results. L/S means the average propagation rate after trending is larger/smaller than before. Then SL/NSL or SS/NSS denotes if it is significantly large/small by t-test. NKS/KS means reject or cannot reject the null hypothesis of KS test.

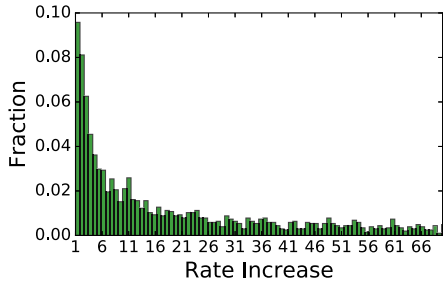


Fig. 2. Distribution of rate increase.

$\frac{\text{avg. retweet delay before trending}}{\text{avg. retweet delay after trending}}$, we obtain the histogram of rate increase in Fig. 2, where the y-axis denote the fraction of topics having rate increase in the range. The rate increase distribution is heavy tailed with some concentration on small numbers (1-10).

Last but not the least, we have a closer inspection of the average retweet delay, grouped by the difference between the retweet times and the corresponding trending times. For example, a -10 means the retweet is 10 hours before the topic get trending. Aggregating more than 60 million retweets from the 3,988 non-trivial trending topics, we obtain the following figure.

The volume of retweets is as expected: most of the retweets are concentrated around the trending time and there are more retweets after trending than before trending. The average retweet delay, however, shows some interesting behavior. First, there are two periods of time that a decrease of the retweet delay is observed: one from 20 hours before trending to the trending time, one around $40-70$ hours after trending. The two periods of decrements are possibly due to different reasons. The earlier one follows our intuition that when a topic starts to be trending, it spreads faster. The latter one may be explained by the hypothesis that the users who are enthusiastic about a topic are likely to keep retweeting it with short delay. The longer after the trending time, the higher the fraction of the enthusiastic users among all users that are still retweeting about the topic. This fact is then reflected in the decrement of retweet delay. Second, the decrease of retweet delay is a gradual process: no single point shows an abrupt drop in delay. This behavior is natural, however, it causes extra complicacy in propagation modeling.

B. The DIP Model

We abstract the OSN as a directed, connected graph $G = (V, E)$, where V denotes all the users in the OSN, and E corresponds to the relationships among the users (follow, friend, etc.) Each edge $(u, v) \in E$ is associated with a weight $p_{uv} \in [0, 1]$ and a probability density function $l_{uv}(\beta)$, which are used to characterize the influence propagation model that is detailed in the following. Also, we consider the propagation rate at time t as $\rho(t)$, which is defaulted at 1 and $\rho(t) > 1$ means a faster propagation.

To model the change in influence propagation rate while considering the impact of social relationship strength, we combine the IC model and the Continuous-Time Diffusion Model into the Continuous Time IC Model (CTIC), whose definition is as follows. We denote the initial set of activated (influenced) nodes as S.

Definition 1 (Continuous Time . Model): Consider a graph $G = (V, E)$ with $l_{uv}(\beta)$ and p_{uv} defined on each edge $(u, v) \in E$. The influence diffusion process starts when all nodes in S are activated at time $t = 0$ and all other nodes remain unactivated. When node u is activated at time t, each neighbor v of u will be activated at time $t + \beta/\rho(t)$ with probability p_{uv} where β follows the probability density function $l_{uv}(\beta)$. Once a node is activated, it will never be deactivated. The process stops when no more nodes can be activated.

Under the CTIC model, we can characterize the rate change using the function $\rho(t)$. The function in general can depict a wide range of propagation scenarios. In this paper, we consider one class of $\rho(t)$ based on the idea in [18] and the findings from Section II-A. Specifically, we consider a topic to be more popular (and thus propagates faster) when the "speedup" event happens: the number of influenced nodes reaches certain thresholds. In the remaining, we will use speedup to denote the event and speedup time to denote the time that the event happens. Based on the findings, the speedups may not be immediate: the propagation rate may gradually increase until reaching the upper bound. We denote $I_t(S)$ as the total number of nodes influenced at time t, given the initial seed set S. Let $0 < \phi_1 < \phi_2 << \phi_\zeta < \phi_{\zeta+1} = 1$ be the thresholds and $1 = r_0 < r_1 < r_2 << r_\zeta, \kappa_1, \kappa_2, \ldots, \kappa_\zeta \geq 0$ as the parameters of speedups. r_is denote the intensity of the propagation rate change and κ_is are used to model the gradual increment in propagation rate between speedup times. Note that the gradual increment can be modeled in many ways as there exists no known results. We select a simple linear model that suffices for the analysis in this paper. It is not the focus of the paper to derive more accurate propagation rate change models. We call the speedup event i happens at time t_i when the following three conditions hold: 1) $I_{t_i}(S) >= \phi_i|V|$. 2) $I_{t_i}(S) < \phi_{i+1}|V|$ and 3)$I_{t_i-\epsilon}(S) < \phi_i|V|$, where $0 < \epsilon < t_i$. For notational convenience, we add a dummy speedup that happens at time $t_{\zeta+1} = +\infty$. Given the speedup times $t_1 \leq t_2 \leq\leq t_{\zeta+1}$, we can then define $\rho(t)$ as follows:

$$\rho(t) = \begin{cases} 1, t < t_1 \\ r_{i-1} + (r_i - r_{i-1})\min\{\frac{t-t_i}{\kappa_i}, 1\}, t_i \leq t < t_{i+1}, i \in [1, \zeta] \end{cases}$$

C. The TAP-DIP Problem.

In a viral marketing campaign, the goal can often be influencing at least a certain number of users within a period of time. For example, a company showcasing its new product will want it to be exposed to a certain percentage of the market within a few days after the release. The company needs to choose some users as seeds to propagate the product information, and seeding each user incurs a cost. For cost-effectiveness, companies always want to minimize the number of seed users, when the costs of seeding the users are the same. Thus, the problem can be rephrased as finding a seed set with minimum size such that the number of activated nodes can be at least a certain threshold $\eta|V|$. Such a problem is termed as the *Threshold Activation Problem (TAP)*. In a typical TAP problem, the underlying influence propagation model is often static, that the parameters of the model will not change overtime. TAP-DIP, however, is the version of TAP that considers dynamic influence propagation models. In this paper, we focus on the TAP problem with the propagation model CTIC.

Definition 2 (TAP-DIP): Given an OSN $G = (V, E)$ with $l_{uv}(\beta)$ and p_{uv} defined on each edge $(u, v) \in E$, the activation threshold η, the trending triggering threshold $\phi_1, \ldots, \phi_\zeta$, the speedup parameters $r_1, \ldots, r_\zeta$ and $\kappa_1, \ldots, \kappa_\zeta$, the time limit T, TAP-DIP asks to find a seed set S with minimum size such that the influence spread $\mathbb{I}_T(S) = E[I_t(S)]$ is at least $\eta|V|$ within time T.

For the majority of the paper, we focus on the case when the propagation rate can change only once and the change is immediate. This case can be obtained from the model discussed above by setting $\zeta = 1$ and $\kappa_1 = 0$. For simplification, we remove the subscripts for ϕ, r and ignore κ in this case. We will discuss the generalized model in Section V.

In the following two sections, we propose FAST, our solution to TAP-DIP with only one possible rate change. For conciseness, most of the proofs are placed in the appendix (available online).

III. FAST: Solution to TAP-DIP

A. Overview

For all existing solutions to influence propagation related problems, a known propagation model is required, which is not possible in TAP-DIP as the propagation rate may change based on number of influenced nodes. To fill the gap, we can provide a key value: the time t that the propagation rate changes. In TAP-DIP, this value is a variable based on number of influenced nodes. When added as an input, it defines fixed propagation models, yet it also brings in the constraint that the number of influenced nodes must meet the triggering threshold ϕ at time t. Thus, with a fixed t, TAP-DIP can be reduced to a problem of finding the minimum seed set to reach $\phi|V|$ and $\eta|V|$ thresholds at time t and T, respectively. We assume that a threshold is met when the expected influence, but not the actual influence, is larger than the threshold as it is not possible to obtain the actual influence when calculating the seed set. Although it may not be exactly the same as in DIP, this assumption is still acceptable: the influence is usually concentrated at the expectation [18]. We also demonstrate in our experiments that the performance of the algorithms are satisfactory, in which we run extensive simulations to demonstrate the performance when we trigger speedups by actual influence. Notice that the propagation models in times $[0, t]$ and $(t, T]$ are different due to the change in propagation rate. We term the problem as Multi-TAP (MTAP), which is a generalization of the TAP problem that only considers satisfying a single threshold $\eta|V|$ at time T, but is still much more accessible than TAP-DIP itself.

As the actual solution to MTAP is complicated, we delay its details in Section IV and assume for now that it is available in a blackbox. We can feed a t value to it and obtain a seed set S_t. Clearly, the solution to TAP-DIP is the S_t with minimum cardinality. However, the function $H(t) = |S_t|$ has no closed form and we have to use global optimization techniques to find its minimum. Since the range of $H(t)$ is discrete, we cannot use tools like Lipschitz optimization [30] directly. Nonetheless, we can define a Lipchitz-alike property for discrete functions and utilize the property to develop our solution, FAST, which can find globally near-optimal values of a function given the property, with limited calls to function value calculation.

Definition 3 (Lipschitz-alike condition for discrete functions): A function $f(x)$ defined on a discrete region X satisfies the Lipschitz-alike condition if there exists a real constant $\mathcal{L} \geq 0$, such that for all $x_1, x_2 \in X$,

$$|f(x_1) - f(x_2)| \leq \mathcal{L}||x_1 - x_2||_2 \tag{1}$$

In the following, we first prove an approximation $H'(t)$ of the function $H(t)$ satisfies (1), and then propose the algorithm FAST that finds near-optimal values of $H(t)$ over t.

B. The FAST Algorithm

If we denote Δ as the minimum distance[3] between any two possible values of t, $H(t)$ is a discrete function and we can prove that $H(t)$ satisfies (1) with $\mathcal{L} = |V|/\Delta$. Unfortunately, the estimation of $\mathcal{L}$ is too crude and it provides little information to minimizing the function. Therefore, we introduce a relaxed version of $H(t)$ that a much smaller $\mathcal{L}$ is achievable.

We write the relaxed version of $H(t)$ as $H'(t) = S^*_s(t) + S^*_a(t)$, where $S^*_s(t)$ denotes the minimum number of nodes to guarantee ϕ fraction of influenced nodes in G on expectation and therefore a speed-up at t; $S^*_a(t)$ denotes the minimum number of nodes to guarantee η fraction of activation in G on expectation *given* a speed-up at t. Notice that $S^*_a(t)$ is calculated based on the assumption that the nodes triggered the speed-up did not influence any nodes in G. Therefore, $H'(t)$ is an upper bound on the number of required nodes, as stated in the following lemma.

Lemma 1: $H'(t^{*'}) \leq 2H(t^*)$ where $t^* = \arg\min_{t\in[0,T]} H(t)$ and $t^{*'} = \arg\min_{t\in[0,T]} H'(t)$.

[3] In practice, we can let t take values $k\Delta, k \in \mathbb{N}$. This is acceptable as we usually use minutes/hours as the smallest time unit for marketing campaigns.

It is clear that $S_s^*(t), S_a^*(t)$ are monotonically decreasing/increasing with t, respectively. Such properties lead to a more accurate estimation of $\mathcal{L}$ values for $H'(t)$.

Lemma 2: Given an interval $[t_1, t_2]$ and function values $H'(t_1) = S_s^*(t_1) + S_a^*(t_1), H'(t_2) = S_s^*(t_2) + S_a^*(t_2)$, $H'(t)$ satisfies (1) over $[t_1, t_2]$ with constant

$$l_{t_1,t_2} = \frac{1}{\Delta}\max\{S_s^*(t_1) - S_s^*(t_2), S_a^*(t_2) - S_a^*(t_1)\} \quad (2)$$

The following lemma adapted from [31] ensures a lower bound on function values within any interval of $H'(t)$.[4]

Lemma 3: When $H'(t)$ satisfies (1) over $[t_1, t_2]$ with constant l_{t_1,t_2},

$$\min_{t\in[t_1,t_2]} H'(t) \geq \frac{H'(t_1) + H'(t_2)}{2} - \frac{l_{t_1,t_2}(t_2 - t_1)}{2} \quad (3)$$

With Lemmas 2 and 3, we propose FAST for finding the global minimum of $H'(t)$ over $[0, T]$, assuming that we have access to the values of $H(t)$ and $H'(t)$, which will be detailed in Section IV.

FAST utilizes the Lipschitz-alike property of $H'(t)$. Intuitively, it iteratively finds the interval with minimum lower bound by Lemma 3 and calculate a new value in the interval, until the interval is small enough. By (1), the minimum of $H'(t)$ is close to one of the calculated values. We further refine the result by calculating $H(\bar{t})$ instead of $H'(\bar{t})$ at the end. The following theorem guarantees the solution quality.

Theorem 1: $H(\bar{t}) \leq 2H(t^*) + 1$, where $t^* = \arg\min_{t\in[0,T]} H(t)$.

Algorithm 1: Finding Anticipated Speedup Time (FAST)

Input: $H'(t), H(t), t \in [0, T]$
Output: The global minimizer $\bar{t}$ of $H'(t)$, $H(\bar{t})$
Calculate $H'(0), H'(T)$ by Algorithm 3.
Let $t_1 = 0, t_2 = T, i, k = 2$, Calculate l_{t_{i-1},t_i} based on (2)
while $|t_i - t_{i-1}| \geq \frac{1}{l_{t_{i-1},t_i}}$ **do**
  $t^{k+1} = \frac{t_i + t_{i-1}}{2} + \frac{H'(t_{i-1}) - H'(t_i)}{2l_{t_{i-1},t_i}}$, k++
  Calculate $H'(t^{k+1})$ by Algorithm 3
  Renumber all points such that $0 \leq t_1 \leq \cdots \leq t_k \leq T$
  for Each interval $[t_{j-1}, t_j]$ **do**
    Calculate l_{t_{j-1},t_j} based on (2) and R_j based on rhs of (3)
  Let $i = \arg\min_{j=1,\ldots,k}\{R_j\}$
$H'(\bar{t}) = \min\{H'(t_i)|i = 1, \ldots, k\}$
$\bar{t} = \arg\min\{H'(t_i)|i = 1, \ldots, k\}$
Calculate $H(\bar{t})$ by Algorithm 3.

IV. MMinSeed: Solution to Multi-Tap

In this section, we describe the missing piece in Section III: how to calculate $H'(t), H(t)$, which completes FAST. As discussed in Section III-A, calculating $H(t)$ is actually solving MTAP with two thresholds. For $H'(t)$, its two components $S_s^*(t), S_a^*(t)$ can be seen as TAP instances with threshold $\theta|V|$ and time limit t, threshold $\eta|V|$ and time limit T, respectively. Therefore, a solution to MTAP suffices for completing FAST. In the following, we propose the first efficient solution to a generalized version of MTAP (defined below) that considers multiple thresholds and time limits. Thus, our proposed solution MMinSeed is not only capable for solving $H(t), H'(t)$, but also applicable to the general scenarios.

Definition 4 (MTAP): Given $G(V, E)$, $V^l \subseteq V, l = 1, \ldots, L$, thresholds $\eta_1, \ldots, \eta_L$, time points $t^1, \ldots, t^L$ and the propagation model, MTAP asks for a seed set S that can influence, by expectation, $\eta_l|V^l|$ nodes in each subset at t^l.

A. The RIS Framework

Due to the complexity from the probabilistic network, sampling is the most popular method to estimate the influence spread of a seed set in each ground set. Here we adopt the state-of-art Reverse Influence Sampling (RIS) technique [9] for generating samples. Specifically, we combine the sampling methods in two recent papers [11], [12] to generate samples for each ground set under Continuous IC propagation model.

The RIS approach has two phases. In the sample generation phase, a number of samples are generated, where each sample consists of all nodes that can influence a random node in a realization of the probabilistic graph. In the seed set selection phase, a maximum coverage problem (with nodes as sets and samples as elements) is greedily solved to obtain the seed set for influence maximization.

The sampling method for TAP-DIP. When the propagation rate increases in TAP-DIP, more nodes may influence the randomly picked node through RIS within the time limit. Hence, it is necessary to extend the RIS sampling method to handle the case. The method we will discuss is not only good for TAP-DIP, but also capable for handling the general case with multiple thresholds and non-immediate change in propagation rates.

Observe that varying the propagation rate is equivalent to varying the time limit: a shorter delay can easily be transformed to the normal delay with the extended time limit. With this observation, we propose an approach to obtain RIS samples with changes in propagation rates. The key idea is, given a set of speedup times, we first convert the time limit under speedups to the time limit with the original propagation rate. Then we proceed with the typical RIS sampling method under the converted time limit. The detail of the time limit conversion algorithm (TLCA) is as follows.

Theorem 2: Obtaining samples with RIS and changes in propagation rates with the original time limits is equivalent to obtaining samples with RIS using the original propagation rate and the time limits outputted by Algorithm 2.

B. The MMinSeed Algorithm

To adapt the RIS framework to solve MTAP, there are two obstacles. The first one is how to guide the solution to consider

[4]The original lemma was for continuous functions, however, the same methodology can be adapted to prove the same result for discrete functions.

Algorithm 2: TLCA

Input: Time limit T, speedup times $t_1, t_2, \ldots, t_\zeta$, speedup parameters $r_1, r_2, \ldots, r_\zeta$ and $\kappa_1, \kappa_2, \ldots, \kappa_\zeta$.
Output: New time limit T'

$T' = t_1$
Let $t_{\zeta+1} = T$, $r_0 = 1$
for i from 1 to ζ **do**
  if $t_{i+1} - t_i > \kappa_i$ **then**
    $T' = T' + \frac{\kappa_i(r_i + r_{i-1})}{2} + (t_{i+1} - t_i - \kappa_i) r_i$
  else
    $T' = T' + \frac{(t_{i+1} - t_i)((t_{i+1} - t_i)(r_{i-1} + r_i) + 2\kappa_i r_{i-1})}{2\kappa_i}$
Return T'

all the thresholds at the same time. A second and more challenging one is that, the RIS framework is designed for maximizing influence with a fixed number of seeds. Also, the number of samples required to guarantee a certain level of accuracy will increase with more seed nodes. However, MTAP asks for minimizing the seed set size, which is unknown and cannot be used to determine the number of required samples.

In order to overcome the first obstacle, we need to design an objective function that satisfies the following conditions: (1) Maximizing the function will fulfill all thresholds. (2) When a threshold is fulfilled, additional influence to the corresponding ground set should not bring any benefit to the function (3) The function must be submodular. The first two conditions insure the correctness of the function, while the last one helps deriving the approximation ratio.

We design the function $f(S)$ as follows:

$$f(S) = \sum_{l \in L} \min\{\eta_l |V^l|, |V^l(S)|\}, \quad S \subseteq \mathcal{S} \tag{4}$$

where $|V^l(S)|$ denotes the expected number of nodes influenced by S in the ground set V^l. Clearly, $f(S)$ is submodular and monotone increasing as it is the sum of submodular functions. Also, each ground set can contribute up to its threshold to the function value. Additionally, the maximum of this function can only be achieved when all thresholds are fulfilled.

We describe MMinSeed in Algorithm 3. In MMinSeed, the process of finding the number of seeds utilizes the submodularity of $f(.)$. In each round, MMinSeed calculates the average gain in $f(.)$ of adding a seed node in the previous round, which is defined as $\overline{\Delta_f} = (f - f_{prev})/(j - j_{prev})$ where f, f_{prev}, j, j_{prev} are the function value $\hat{f}(S)$ and number of seeds in the previous two rounds, respectively. Then, the algorithm estimates how many new seed nodes are required, denoted by Δ_j, assuming all the new nodes can bring the gain equal to $\overline{\Delta_f}$. The approach will reduce the number of calls to its subroutine, Algorithm 4. Comparing with binary search, the greatest advantage of this approach is that it will never choose a seed set size that is larger than necessary, which is guaranteed by submodularity. A larger seed set is not preferable since it leads to generating more samples, which is redundant and costs extra time.

The subroutine Multi-IM (Algorithm 4), is the first efficient solution to the MIM problem. It maintains L collections of

Algorithm 3: MMinSeed

Input: Graph $G = (V, E)$, Ground sets $V^1, \ldots, V^L$ with thresholds $\eta_1, \ldots, \eta_L$, $\epsilon > 0$
Output: Seed set $S \subseteq V$

$f(S^*) = \sum_{l \in L} \eta_l |V^l|$.
$j = 1, j_{prev} = 0, f = 0, f_{prev} = 0, S = \emptyset$
Find S using Algorithm 4 with $|S| \leq j$
$f = \hat{f}(S)$
while $\hat{f}(S) < (1 - \epsilon) f(S^*)$ **do**
  $\overline{\Delta_f} = (f - f_{prev})/(j - j_{prev})$
  $\Delta_j = \lceil \frac{(1-\epsilon) f(S^*) - \hat{f}(S)}{\Delta(f)} \rceil$
  $f_{prev} = f, j_{prev} = j$
  $j = j + \Delta_j$
  Find S using Algorithm 4 with $|S| \leq j$
  $f = \hat{f}(S)$

Algorithm 4: Multi-IM

Input: Graph $G = (V, E)$, Ground sets $V^1, \ldots, V^L$ with thresholds $\eta_1, \ldots, \eta_L$, Precision parameters $\epsilon > 0, \delta \in (0, 1)$
Output: Seed set $S \subseteq V$

Collection of samples $\mathcal{R}^l = \emptyset, l = 1, \ldots, L$
$\phi^l = \frac{(1-1/e)\sigma + \tau^l}{\epsilon}$, $\gamma^l = \frac{7e(\phi^l)^2}{3(e-1)}$
$N_{\mathcal{R}^l} = \gamma^l, ctn^l = true, l = 1, \ldots, L$
while $\exists l$ s.t. $ctn^l == false$ **do**
  S_k = Greedy size k solution to maximize $\hat{f}(S)$
  for Each $l = 1, \ldots, L$ **do**
    if ctn^l **then**
      if $C_{\mathcal{R}^l}(S_k) \geq \gamma^l$ **then**
        $ctn^l = false$
      else
        Generate $N_{\mathcal{R}^l}$ samples for $\mathcal{R}^l$, $N_{\mathcal{R}^l} = 2N_{\mathcal{R}^l}$

samples $\mathcal{R}^1, \ldots, \mathcal{R}^l$ (one for each threshold) and it keeps generating new samples for each collection up to a given amount $N_{\mathcal{R}^l}$. Then, Algorithm 4 greedily solves a submodular maximization problem with the submodular function $f(S)$ defined in (4), using at most k nodes. The resulting set S_k is used to verify if the number of samples intersect with S_k, $C_{\mathcal{R}^l}(S_k)$, is at least γ^l. If the verification is successful, $\mathcal{R}^l$ is enough to guarantee the accuracy of estimating the ground set. It then stops generating new samples for $\mathcal{R}^l$. Otherwise, it doubles $N_{\mathcal{R}^l}$, generating samples up to $N_{\mathcal{R}^l}$ and rerun the verification. When all $\mathcal{R}^l$ passed the verification, the solution S_k is returned as output.

C. Theoretical Analysis.

Since $f(.)$ is submodular, the following result [32] holds using the greedy algorithm, if all $f(.)$ values can be obtained in polynomial time:

$$f(S_j^g) \geq (1 - (1 - 1/k)^j) f(S_k^*)$$

where S_j^g is the collection of the first j sets selected by the greedy algorithm to maximize $f(.)$ and S_k^* is the optimal collection of size k. However, in Multi-IM, the values of $f(.)$ are not accurate but estimated by RIS. Hence, we can only have a weaker result as in Theorem 3.

Theorem 3: Algorithm 4 guarantees

$$f(S_j^g) \geq (1-(1-1/k)^j - \epsilon) f(S_k^*) \tag{5}$$

with probability at least $1-\delta$.

To prove Theorem 3, we prove the following two Lemmas. In lemma 4, we derive the number of samples required to guarantee (5). Next, we ensure that Algorithm 4 generates at least that many samples in Lemma 5. The validity of Theorem 3 is then derived by combining Lemma 4 and Lemma 5, and applying the union bound. Notice that $L \in \mathbb{N}^+$ and $\delta \in (0,1)$, so the probabilities in Lemma 4 and Lemma 5 are well defined.

Lemma 4: The number of samples required to guarantee (5) with probability at least $1-\frac{L+1}{2L+1}\delta$ is

$$Q = \sum_{l=1}^{L} Q^l \tag{6}$$

$$Q^l = \frac{7e|V^l|(\phi^l)^2}{3(e-1)\mathbb{I}_T^l(S_k^*)}, \phi^l = \frac{(1-1/e)\sigma + \tau^l}{\epsilon}$$

$$\sigma = \sqrt{\ln(\frac{2L+1}{\delta})}, \tau^l = \sqrt{(1-\frac{1}{e})(\ln\frac{(2L+1)\binom{|V^l|}{j}}{\delta})}$$

$\delta \in (0,1)$ and $\epsilon \in (0, \frac{1}{2})$ are constants.

Lemma 5: Algorithm 4 guarantees the number of samples for each threshold is at least Q^l when it stops, with probability at least $1-\frac{L}{2L+1}\delta$.

It is possible that $\mathbb{I}_T^l(S_k^*)$ is close to 0. This will only happen if the probability p_{uv} for all edges $(u,v) \in E$ are close to 0. We ignore such cases since no strategy can grant any benefit.

With Theorem 3, we are able to derive the approximation ratio of MMinSeed in Theorem 4 and eventually, the approximation ratio of FAST in Theorem 5.

Theorem 4: MMinSeed has approximation ratio $\log|V|$ and achieves at least $(1-\epsilon)$ of the required $f(.)$ value, given (5).

Theorem 5: FAST has approximation ratio of $2\log|V|$.

Theorem 6: The time complexity of FAST is $O(\frac{mnT}{\Delta}\sum_{l\in L}(\phi^l)^2))$, where n, m are number of nodes and edges in the graph, respectively.

V. Solution to the General Problem

In this section, we lift the constraints on ζ and κ. In other words, there may be multiple speedup events and they may not be immediate. In this general problem, the main challenge of TAP-DIP remains and is more complicated: we need to have the knowledge of multiple, instead of one, speedup times to enable sampling. It requires a non-trivial extension to Algorithm 1.

Following the idea in Algorithm 1, we would like to find the best set of speedup times that results in the minimum number of seeds. Doing so requires optimizing a multi-variate function $H(t_1, t_2, \ldots, t_\zeta)$ (or $H(\mathbf{t})$ in short) and Algorithm 1, designed for univariate function minimization, cannot work. However, we can generalize the approach. Recall that Algorithm 1 iteratively finds a new evaluation point within an interval that is having the least lower bound on function value and then separate the interval to two, update the l values and continue. With ζ dimensions, we need a more complicated data structure to organize the regions: we encode the regions in a hyperoctree [33], in which each region is a node in tree and a non-leaf node (corresponds to a separated region) will have exactly 2^ζ children. A region is defined by two ζ dimensional vectors $\mathbf{t}^l = (t_1^l, \ldots, t_\zeta^l)$ and $\mathbf{t}^h = (t_1^h, \ldots, t_\zeta^h)$, with $t_i^l < t_i^h, \forall i = 1, \ldots, \zeta$. We term them as the defining vectors of the region. The vertices of the region will be all ζ dimensional binary vectors, where a 0 in position i indicates the vertex has value t_i^l for dimension i, and 1 otherwise. With a new evaluation point $\mathbf{t}'$, the separation will take place in the region that satisfies $t_i^l < t_i' < t_i^h, \forall i = 1, \ldots, \zeta$.[5] Each one of the 2^ζ new regions will involve a vertex $\mathbf{t}^o$ of the old region and the point $\mathbf{t}'$. The two defining vectors for the new region are the pairwise minimum/maximum of the vectors $\mathbf{t}^o$ and $\mathbf{t}'$, respectively.

When the regions are available, we may generalize Lemma 1 and Lemma 2 to calculate the l values for each region. Let $S_{s_i}^*(t)$ be the number of seeds required to trigger the ith speedup event *without* considering the impact of any other speedups and $S_a^*(\mathbf{t})$ be the number of seeds required to guarantee η fraction of activation in G given speeds happening at times $\mathbf{t} = (t_1, t_2, \ldots, t_\zeta)$. Let $H'(\mathbf{t}) = \sum_{i=1}^{\zeta} S_{s_i}^*(t_i) + S_a^*(\mathbf{t})$, we have the following results,[6]:

Lemma 6: $H'(\mathbf{t}^{*'}) \leq (\zeta+1)H(\mathbf{t}^*)$ where $\mathbf{t}^{*'}, \mathbf{t}^*$ are minimizers for $H'(.)$ and $H(.)$, respectively.

Lemma 7: Given the two defining vectors $\mathbf{t}^l = (t_1^l, \ldots, t_\zeta^l)$ and $\mathbf{t}^h = (t_1^h, \ldots, t_\zeta^h)$ of a region and function values $H'(\mathbf{t}^l), H'(\mathbf{t}^h)$, let Δ be the minimum distance between any two possible values of $\mathbf{t}$, then $H'(\mathbf{t})$ satisfies (1) in the region with constant

$$l_{\mathbf{t}^l,\mathbf{t}^h} = \frac{1}{\Delta}\max\{S_a^*(\mathbf{t}^l) - S_a^*(\mathbf{t}^h), \sum_{i=1}^{\zeta} S_{s_i}^*(t_i^l) - \sum_{i=1}^{\zeta} S_{s_i}^*(t_i^h)\} \tag{7}$$

With $l_{\mathbf{t}^l,\mathbf{t}^h}$ values and function values $H'(\mathbf{t}^l), H'(\mathbf{t}^l)$, we can obtain a lower bound for all function values when the variables are in the region.

Lemma 8: The lower bound $R_{\mathbf{t}^l,\mathbf{t}^h}$ of function values for the region defined by $\mathbf{t}^l, \mathbf{t}^h$ satisfies:

$$R_{\mathbf{t}^l,\mathbf{t}^h} \geq \frac{H'(\mathbf{t}^l) + H'(\mathbf{t}^h)}{2} - \frac{l_{\mathbf{t}^l,\mathbf{t}^h}||\mathbf{t}^h - \mathbf{t}^l||_2}{2} \tag{8}$$

In a region, the points $\mathbf{t}'$ that are likely to hit the lower bound satisfy

$$\frac{H'(\mathbf{t}^h) - H'(\mathbf{t}^l)}{l_{\mathbf{t}^l,\mathbf{t}^h}} = ||\mathbf{t}^h - \mathbf{t}'||_2 - ||\mathbf{t}' - \mathbf{t}^l||_2 \tag{9}$$

[5]We omit the possible equality here, as it will just result in a degenerated problem with less regions per separation.

[6]The proofs are similar to those for Lemma 1 2 and thus omitted.

Algorithm 5: Finding Anticipated Speedup Times (FASTS)

Input: $H'(\mathbf{t}), H(\mathbf{t}), \in [\mathbf{0}, \mathbf{T}]^{\zeta}$
Output: The global minimizer $\mathbf{t}^*$ of $H'(\mathbf{t}), H(\mathbf{t}^*)$
Calculate $H'(\mathbf{0}), H'(\mathbf{T})$ by Algorithm 3.
Construct a tree with the region $H'(\mathbf{0}), H'(\mathbf{T})$ as the root.
Let the min region $\mathbb{R}_{min} = (H'(\mathbf{0}), H'(\mathbf{T})), P = \{\mathbf{0}, \mathbf{T}\}$.
Calculate $l_{\mathbf{0},\mathbf{T}}$ based on (7)
while $\mathbb{R}_{min}$ $(\mathbf{t}^l, \mathbf{t}^h)$ satisfies $||\mathbf{t}^l - \mathbf{t}^h||_2 \geq \frac{1}{l_{\mathbf{t}^l,\mathbf{t}^h}}$ **do**
  Find the new evaluation point $\mathbf{t}'$ in $\mathbb{R}_{min}$ based on (10).
  Calculate $H'(\mathbf{t}')$ by Algorithm 3
  Add $\mathbf{t}'$ to P.
  Construct 2^{ζ} new regions as children of $\mathbb{R}_{min}$ in the tree.
  for Each region as a leaf in the tree **do**
    Calculate $l_{\mathbf{t}^l,\mathbf{t}^h}$ based on (7) and $R_{\mathbf{t}^l,\mathbf{t}^h}$ based on (8).
  Let $\mathbb{R}_{min} = \arg\min_{\text{region } (\mathbf{t}^l,\mathbf{t}^h) \text{ as leaves in the tree}} \{R_{(\mathbf{t}^l,\mathbf{t}^h)}\}$
$H'(\mathbf{t}^*) = \min\{H'(\mathbf{t}')|\mathbf{t}' \in P\}, \mathbf{t}^* = \arg\min\{H'(\mathbf{t}')|\mathbf{t}' \in P\}$
Calculate $H(\mathbf{t}^*)$ by Algorithm 3.

As there can be infinitely many such points when $\zeta \geq 2$ (e.g. the points form a hyperbola with $\zeta = 2$), we consider the only point that is on the segment connecting $\mathbf{t}^l, \mathbf{t}^h$. Combining (9) and the fact that the point is on the segment, we solve the point as follows:

$$\mathbf{t}' = (\alpha t_1^h + (1-\alpha)t_1^l, \ldots, \alpha t_\zeta^h + (1-\alpha)t_\zeta^l)$$
$$\text{where } \alpha = 1 - \left(\frac{|H'(\mathbf{t}^h) - H'(\mathbf{t}^l)|}{2l_{\mathbf{t}^l,\mathbf{t}^h}||\mathbf{t}^l - \mathbf{t}^h||_2} + \frac{1}{2}\right)^2 \quad (10)$$

With all the above analysis, we have Algorithm 5 that solves our general problem. Notice that with a fixed set of $t_1, \ldots, t_\zeta$, Algorithm 3 and Algorithm 4 are capable of finding the minimum number of seeds, even in this general case.

Theorem 7: $H(\mathbf{t}^*) \leq (\zeta+1)H(\mathbf{t}^{OPT}+1)$

The proof for the above performance guarantee is an extension to that of Theorem 5 and hence omitted.

Theorem 8: The time complexity of FASTS is $O(mn\left(\frac{T}{\Delta}\right)\sum_{l\in L}(\phi^l)^2))$.

VI. Experiments

A. Experimental Settings.

The experiments are conducted on a Linux machine with 2.3 GHz Xeon 18 core processor and 256 GB of RAM. We carry experiments under Continuous IC models on the following datasets from [34].

Datasets. We select 8 OSN datasets of various sizes to test the impact of the dynamic influence propagation model. The description summary of those datasets is shown in Table I.

Parameter Settings. We follow the papers [6], [10] for setting propagation probability p_{uv}, which is calculated as $p_{uv} = \frac{1}{d_v^{in}}$ where d_v^{in} denotes the indegree of node v. We model the propagation rate using Weibull distribution as [7], [11] and fix the shape parameter at 4, scale parameter at 1 throughout the experiments.

TABLE I
Datasets' Statistics

Dataset	#Nodes	#Edges	T-Node	A-Node
Facebook	4K	88K	100 - 500	1K - 2K
wiki-Vote	7K	206K	100 - 500	1K - 2K
Epinions	76K	1M	500 - 2.5K	10K - 20K
Slashdot	77K	1.8M	500 - 2.5K	10K - 20K
Twitter	81K	3.54M	500 - 2.5K	10K - 20K
Gplus	108K	26M	500 - 2.5K	10K - 20K
Pokec	1.63M	61.2M	5K-25K	100K-200K
LiveJournal	4.85M	138M	5K-25K	100K-200K

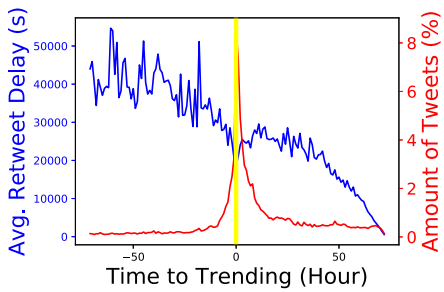


Fig. 3. Retweet delays.

In all the experiments, we keep $\epsilon = 0.1$ and $\delta = 1/n$ if the values are not stated otherwise. The time limit is set at 10. The values of r, the propagation rate change, varies from 1.5 to 4.0. The ϕ, η values are not set explicitly, instead, we set the number of nodes required for being trending (T-Node) and for overall activation requirement(A-Node), based on the size of the networks. The parameters are summarized in Table I.

B. Performance of FAST.

Since the TAP-DIP problem is new, there are no suitable algorithms that can be compared directly with FAST. Instead, we demonstrate the performance of FAST via its subroutine. We compare the MMinSeed algorithm with its variation that uses the IMM algorithm [11] instead of the Multi-IM algorithm (Algorithm 4) as a subroutine. In the comparison, we only allow one threshold and modify IMM's sampling method to allow the Continuous-IC model. The comparison is based on the scenario with no rate change and the lowest activation threshold for each network.

Figure 4 proves a clear difference of the running time.The Multi-IM supported MMinSeed is much faster (the running time is in log scale) than the one supported by IMM. The main reason for the superior performance of Multi-IM is that it decides the sample requirement dynamically, while IMM has a parameter estimation stage to estimate the number of required samples, which can be inaccurate and results in a sample collection that is much larger than necessary when the seed set size is small. In the running time comparison, we set $\epsilon = 0.5$ to allow IMM finish in reasonable time. Each number is the average of 10 runs, as the variation of running time is small and the difference is apparent.

Scalability of FAST. During our experiments, we observe that all the scenarios had less than 20 iterations inside FAST, which means the time complexity of FAST is larger than that of

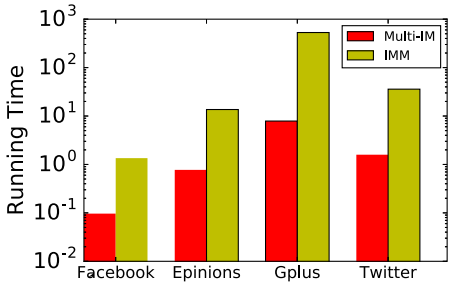


Fig. 4. Efficiency of MMinSeed with Multi-IM/IMM.

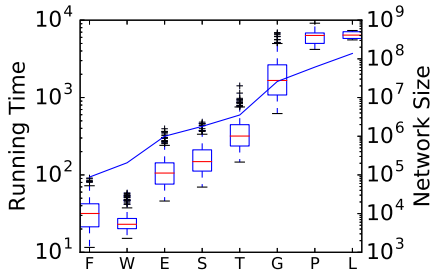


Fig. 5. Running Time of FAST.

MMinSeed only by a multiplicative constant (< 40). Figure 5 demonstrates the scalability of FAST. The trend line (corresponds to the right y-axis) denotes the size of different networks (number of edges) and the box plots (corresponds to the left y-axis) displays the running time in different networks with various settings. Notice that we use the first letter of each dataset due to space issues. We can observe a nice feature of FAST that its running time grows linearly in terms of network size. For large networks such as Pokec (61.2 M edges) and LiveJournal (138 M edges), FAST can finish within three hours.

C. Quality of the Seed Sets.

One key factor we would like to consider is how the seed set will differ with/without the consideration of DIP. Figure 6 compares the seed sets obtained by FAST and those obtained by solving the base case of TAP without considering the rate increase. In the pie charts, the shared seeds means the proportion of seed nodes that exists in both FAST seeds and base seeds. The data in each chart is averaged among the result from 180 runs of various settings. Clearly, considering the rate increase explicitly result in a large reduction in number of seeds required.

As the FAST seed set is mostly a subset of the base seed set, we would expect that the FAST seed set has a smaller influence spread. However, we will demonstrate that the FAST seeds can still meet the threshold with extensive simulations. For each setting in each dataset, we simulate 10,000 random propagation process from both the FAST seeds and the base seeds. The rate increase is applied when the *actual* number of influenced nodes hit the trending thresholds. Figure 7 depicts the percentage of nodes activated comparing with the corresponding threshold. As we set ϵ at 0.1, the reference line (in yellow) is drawn at 90%. A point over the reference line means the average number of activated nodes (over 10,000 simulations) in a certain scenario meets 90% of the threshold. In all the four datasets, the base seed set activated much more nodes than required by the threshold, which suggests that their size can actually be reduced. In most cases, the FAST seed sets can meet the requirement. One exception is in the LiveJournal data set, it is possible that the influence spread in LiveJournal is not concentrated at the expectation.

D. Sensitivity Analysis of Key Parameters

In this section, we test the impact of three parameters: r (the propagation rate after trending), T-Node (# nodes required to be trending) and A-Node (overall activation requirement) to

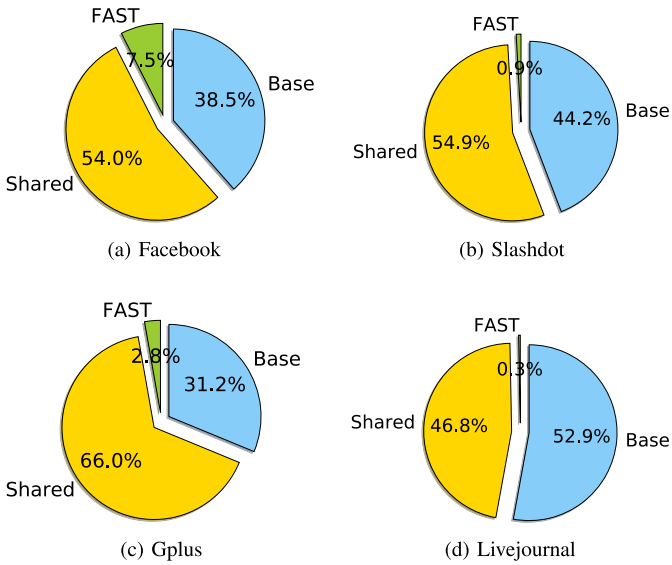
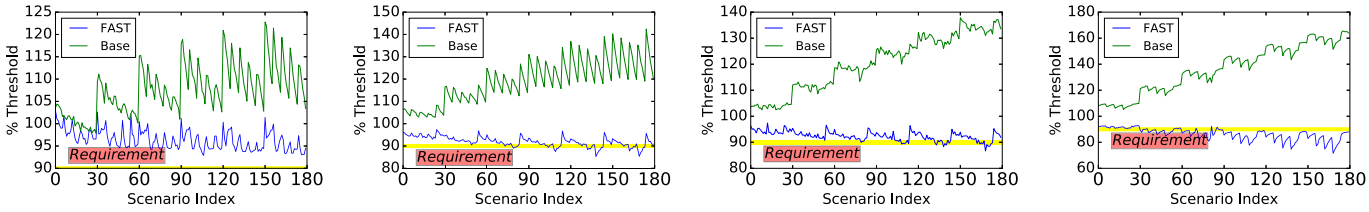


Fig. 6. Seed Set Distribution.

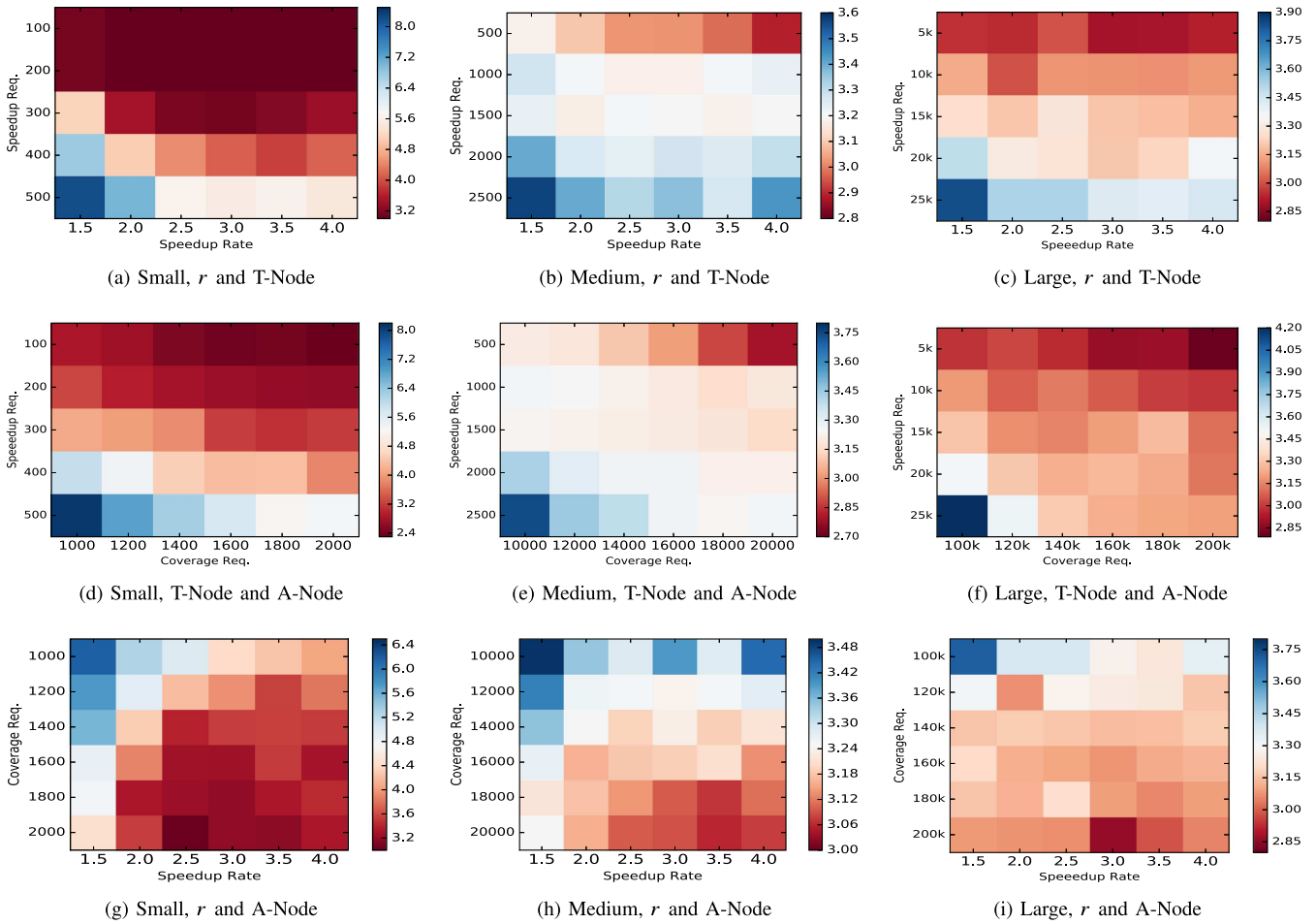
the behavior of FAST. The results are displayed in heat maps, a warmer color means an earlier time for rate increase (decided by FAST). The data sets are grouped based on parameter setting (refer to Table I for details) into small (Facebook, WikiVote), medium (Gplus, Twitter, Epinions, Slashdot) and large (LiveJournal, Pokec). In each heat map, we vary two parameters and take average over the other. A warmer color in a cell denotes a earlier time for rate increase.In Fig. 8a-8c, we vary r and T-Nodes. One clear trend is that when more T-Nodes are required, FAST tends to choose a later time for starting the trend and the speed up. It is reasonable as it can be costly to activate T-Nodes early when the number is high. In some scenarios, FAST may choose not to have the speed up at all when T-Node is too large and r is small. What seems counter-intuitive in those Figures is that FAST may not choose to have the speed up earlier when r increases. This phenomena can possibly explained from the perspective of cost. With a larger r, FAST can influence a set of nodes in less time, even with the same seed set. Thus, FAST is not that "hurry" of starting the speed up, as it is able to reach the threshold when triggering the speed up later with less cost.

In Figs. 8d-8i, we vary A-Nodes. In most of the cases, FAST tend to have the speed up earlier when facing a larger threshold. The opposite is shown in some rare cases, mostly because r is too low or T-Node is too high, which makes FAST think starting the speed-up early is not beneficial.



(a) Facebook (b) Slashdot (c) Gplus (d) LiveJournal

Fig. 7. Percentage of Nodes Activated.



(a) Small, r and T-Node (b) Medium, r and T-Node (c) Large, r and T-Node

(d) Small, T-Node and A-Node (e) Medium, T-Node and A-Node (f) Large, T-Node and A-Node

(g) Small, r and A-Node (h) Medium, r and A-Node (i) Large, r and A-Node

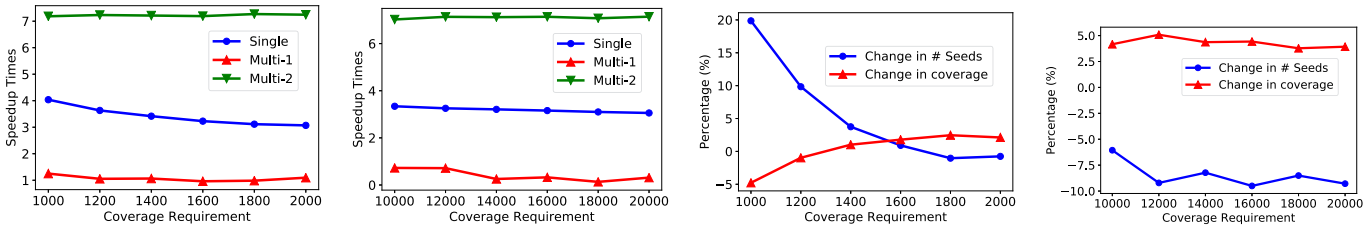
Fig. 8. Sensitivity Analysis Results.

E. Result for the General Problem

The main interest we have for the general problem is: how the results differ from the case with only one speedup event? Hence, we create the general scenarios based on single speedup scenarios. For a single speedup scenario with speedup rate r and trending requirement (T-Node) N_T, we apply the following changes to obtain a general scenario: 1) Add a speedup that happens with threshold $\frac{N_T}{2}$ and rate $\frac{r+1}{2}$. 2) For the two speedups, let $\kappa = 1$. Notice that we keep the overall activation requirement (A-Node) the same.

We run both the single speedup and the general scenario in Facebook and Slashdot datasets. For the running time, the general scenario usually runs around 10 times slower than the single speedup scenario. This is understandable. Each time when we evaluate a new node in the single speedup scenario, we only need to run Algorithm 3 for that node. In the general scenario, however, we need to do so for both defining vectors of all the 2^κ (4 in our experiment as we have $\kappa = 2$) new regions. Also, each call to Algorithm 4 requires an extra set of samples for the new speedup threshold. On the other hand, the convergence of Algorithm 5 is similar to that of Algorithm 1, each call to Algorithm 5 ends within 30 iterations of the main loop.

In Figs. 9(a) and 9(b), we compare the speedup times in both scenarios. It is interesting that Algorithm 5 decided the first speedup in the general case should be very early and the second should be very late. Having the first speedup can be very



(a) Speedup times in Facebook (b) Speedup times in Slashdot (c) Change in # of seeds and coverage in Facebook (d) Change in # of seeds and coverage in Slashdot

Fig. 9. Comparison between single/multiple speedups.

tempting as the requirement is low (only half of the second). However, as we calculate the seeds for speedups separately, the cost to have the second speedup early can be high. With the first speedup available, the need for a early second speedup is not strong. It is costly but without much benefit, especially considering the fact that the speedup is not immediate.

We then illustrate the impact of the general case in regards to the size and the quality (represented by the *actual* coverage in simulation) of the seeds. The changes are calculated as

$$\text{Change in } \#\text{ seeds} = \frac{\#\text{ seeds in single - }\#\text{ seeds in general}}{\#\text{ seeds in single}}$$

$$\text{Change in cov.} = \frac{\text{act. cov. in general - act. cov. in single}}{\text{act. cov. in single}}$$

In Fig. 9c, we can observe a clear reduction in # seeds when the activation (coverage) requirement is low. As depicted in Fig. 9a, the first speedup is usually triggered early. In a small network like Facebook, doing so does not require a large number of seeds and it is not necessary to have many more seeds to meet the coverage requirement. However, the actual coverage was slightly reduced, yet it was still above the 90% coverage requirement, as desired. When coverage requirement is 1400 and 1600, the result for the general scenario shows improvement on both ends: less seeds, more coverage. With higher activation requirement, however, the number of seeds increased. The main reason may be the slightly reduced approximation ratio (from $2\log|V|$ to $3\log|V|$) and the selection of two speedup times. The algorithm will need more seeds to ensure the coverage requirement to remedy the fact that the second speed up happens really late and not immediate, comparing with the single speed up scenario. The same analysis extends to Fig. 9d.

VII. Conclusion

In this paper, we proposed a novel dynamic influence propagation model, which can more accurately characterize the information diffusion in social networks compared with the existing propagation models. The model is supported by analysis of the crawled retweet data, that most topics will propagate faster after being trending. To study the impact of DIP in OSNs, we propose a new TAP-DIP problem by substituting the static propagation model in TAP with DIP. Although TAP-DIP is even harder than TAP, we designed the FAST algorithm that can solve it with approximation ratio similar to the best for TAP. We also propose a solution to the general TAP-DIP problem, which allows multiple rate changes, with similar guarantee. In experiments, we demonstrated that FAST can generate high quality seed sets and is scalable. Also, we confirmed that DIP has a high impact in the result.

References

[1] P. Domingos and S. Richardson, "Mining the network value of customers," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, ACM, 2001, pp. 57–66.
[2] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, ACM, 2002, pp. 61–70.
[3] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 137–146.
[4] J. Leskovec, A. Krause, C Guestrin, C. Faloutsos, J VanBriesen, and N Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 420–429.
[5] W. Chen, C. Wang, and Y Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 1029–1038.
[6] A. Goyal, W. Lu, and L. VS Lakshmanan, "Simpath: An efficient algorithm for influence maximization under the linear threshold model," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 211–220.
[7] N. Du, L. Song, M. Gomez-Rodriguez, and H Zha, "Scalable influence estimation in continuous-time diffusion networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2013, pp. 3147–3155.
[8] E. Cohen, D. Delling, T Pajor, and R. F. Werneck, "Sketch-based influence maximization and computation: Scaling up with guarantees," in *Proc. 23 rd ACM Int. Conf. Inf. Knowl. Manag.*, 2014, pp. 629–638.
[9] C. Borgs, M. Brautbar, J Chayes, and B Lucier, "Maximizing social influence in nearly optimal time," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2014, pp. 946–957.
[10] Y. Tang, X. Xiao, and Y Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. ACM SIGMOD Int. Conf. Manag. data*, 2014, pp. 75–86.
[11] Y. Tang, Y. Shi, and X Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2015, pp. 1539–1554.
[12] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Cost-aware targeted viral marketing in billion-scale networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2016, pp. 1–9.
[13] X Li, J. D Smith, T. N. Dinh, and M. T. Thai, "Why approximate when you can get the exact? optimal targeted viral marketing at scale," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2017, pp. 1–9.
[14] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. Int. Conf. Manag. Data*, 2016, pp. 695–710.
[15] C. Long and R. C.-W. Wong, "Minimizing seed set for viral marketing," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 427–436.
[16] A. Goyal, F. Bonchi, L. VS Lakshmanan, and S. Venkatasubramanian, "On minimizing budget and time in influence propagation over social networks," *Social Netw. Anal. Mining*, vol. 3, no. 2, pp. 179–192, 2013.
[17] D. T. Nguyen, H Zhang, S Das, M. T. Thai, and T. N. Dinh, "Least cost influence in multiplex social networks: Model representation and analysis," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 567–576.

[18] P. Zhang, W. Chen, X Sun, Y Wang, and J Zhang, "Minimizing seed set selection with probabilistic coverage guarantee in a social network," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 1306–1315.
[19] T. N. Dinh, H. Zhang, D. T. Nguyen, and M. T. Thai, "Cost-effective viral marketing for time-critical campaigns in large-scale social networks," *IEEE/ACM Trans. Netw.*, vol. 22, no. 6, pp. 2001–2011, Dec. 2014.
[20] A. Kuhnle, T. Pan, M. A. Alim, and M. T. Thai, "Scalable bicriteria algorithms for the threshold activation problem in online social networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2017, pp. 1–9.
[21] T. Pan, A. Kuhnle, X Li, and M. T. Thai, "Dynamic propagation rates: New dimension to viral marketing in online social networks," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 1021–1026.
[22] A Kuhnle, M. A Alim, X Li, H Zhang, and M. T. Thai, "Multiplex influence maximization in online social networks with heterogeneous diffusion models," *IEEE Trans. Comput. Social Syst.,* vol. 5, no. 2, pp. 418–429, Jun. 2018.
[23] L. Sun, W. Huang, P. S. Yu, and W Chen, "Multi-round influence maximization," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2249–2258.
[24] X. He and D. Kempe, "Stability and robustness in influence maximization," *ACM Trans. Knowl. Discovery Data*, vol. 12, no. 6, p. 66, pp. 1–34, 2018.
[25] C. V. Pham, H. V. Duong, H. X. Hoang, and M. T. Thai, "Competitive influence maximization within time and budget constraints in online social networks: An algorithmic approach," *Appl. Sci.*, vol. 9, no. 11, pp. 2274–2302, 2019.
[26] S. Aral and P. S. Dhillon, "Social influence maximization under empirical influence models," *Nature Human Behaviour*, vol. 2, no. 6, pp. 375–382, 2018.
[27] X Li, J D Smith, T. N. Dinh, and M. T. Thai, "Tiptop: Almost exact solutions for influence maximization in billion-scale networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 649–661, Apr. 2019.
[28] L. N. Nguyen, K Zhou, and M. T. Thai, "Influence maximization at community level: A new challenge with non-submodularity," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 327–337.
[29] M. H. DeGroot and M. J. Schervish, "Kolmogorov–smirnov tests," *Probability and Statistics. London, U. K.: Pearson*, 2011, pp. 657–58.
[30] R. Horst and P. M. Pardalos. *Handbook of Global Optimization*, vol. 2. Berlin, Germany: Springer, 2013.
[31] Daniela Lera and Yaroslav D Sergeyev, "Acceleration of univariate global optimization algorithms working with lipschitz functions and lipschitz first derivatives," *SIAM J. Optim.*, vol. 23, no. 1, pp. 508–529, 2013.
[32] T. Elomaa and J. Kujala, "Covering analysis of the greedy algorithm for partial cover," in *Algorithms and Applications*, Berlin, Germany: Springer, 2010, pp. 102–113.
[33] M.-M. Yau and S. N. Srihari, "A hierarchical data structure for multidimensional digital images," *Commun. ACM*, vol. 26, no. 7, pp. 504–515, 1983.
[34] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," 2014. [Online]. Available: http://snap.stanford.edu/data

Tianyi Pan received the Ph.D. degree in computer engineering from the University of Florida, Gainesville, FL, USA. His research focuses on approximation algorithms of optimization problems and vulnerability analysis in interdependent networks, including online social networks, smart grid and communication networks.

Xiang Li (Member, IEEE) received the Ph.D. degree from the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL, USA. She is currently an Assistant Professor with the Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA, USA. Her research interests are centered on the large-scale optimization and its intersection with cyber-security of networking systems, big data analysis, and cyber physical systems. She has authored or coauthored 25 articles in various prestigious journals and conferences, such as the IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON SMART GRIDS, IEEE INFOCOM, IEEE ICDM, including one Best Paper Award in IEEE MSN 2014, Best Paper Nominee in IEEE ICDCS 2017, and Best Paper Award in IEEE International Symposium on Security and Privacy in Social Networks and Big Data 2018. She is an Associate Editor for the *Computational Social Networks* journal and the *Journal of Combinatorial Optimization*.

Alan Kuhnle (Member, IEEE) received the M.Sc. degree in mathematics and the Ph.D. degree in computer science both from the University of Florida, Gainesville, FL, USA. He is currently an Assistant Professor with the Department of Computer Science, Florida State University. His current research focuses on the design and analysis of algorithms to solve optimization problems arising from machine learning, network science, and bioinformatics. Particular interests currently include scalable optimization of submodular functions, fast evolutionary algorithms with theoretical guarantees and succinct data structures for representing genomics data.

My T. Thai (Member, IEEE) is currently a UF Research Foundation Professor with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL, USA. Her current research interests are on scalable algorithms, big data analysis, cybersecurity, and optimization in network science and engineering, including communication networks, smart grids, social networks, and their interdependency. The results of her work have led to six books and more than 200 articles, including IEEE MSN 2014 Best Paper Award, 2017 IEEE ICDM Best Papers Award, 2017 IEEE ICDCS Best Paper Nominee, and 2018 IEEE/ACM ASONAM Best Paper Runner up.

She has engaged in many professional activities. She has been a TPC-chair for many IEEE conferences, is a founding EiC of the *Computational Social Networks* journal, EiC of the *Journal of Combinatorial Optimization*, and a book series editor of *Springer Briefs* in *Optimization* and Springer Optimization and its Application. She has received many research awards, including a UF Provosts Excellence Award for Assistant Professors, UFRF Professorship Award, a Department of Defense (DoD) Young Investigator Award, and an NSF (National Science Foundation) CAREER Award.