Modeling Non-Progressive Phenomena for Influence Propagation

Vincent Yun Lou
Stanford University
yunlou@stanford.edu

Smriti Bhagat
Technicolor
smriti.bhagat@technicolor.com

Laks V.S. Lakshmanan   Sharan Vaswani
University of British Columbia
{laks,sharanv}@cs.ubc.ca

ABSTRACT

Recent work on modeling influence propagation focus on progressive models, i.e., once a node is influenced (active) the node stays in that state and cannot become inactive. However, this assumption in unrealistic in many settings where nodes can transition between active and inactive states. For instance, a user of a social network may stop using an app and become inactive, but again activate when instigated by a friend, or when the app adds a new feature or releases a new version. In this work, we study such non-progressive phenomena and propose an efficient model of influence propagation. Specifically, we model influence propagation as a continuous-time Markov process with 2 states: active and inactive. Such a model is both highly scalable (we evaluated on graphs with over 2 million nodes), 17-20 times faster, and more accurate for estimating the spread of influence, as compared with state-of-the-art progressive models for several applications where nodes may switch states.

1. INTRODUCTION

Study of information and influence propagation over social networks has attracted significant research interest over the past decade, driven by applications such as viral marketing [19, 10], social feed ranking [30], contamination detection [21, 26, 1], and spread of innovation [31] to name a few. A prototypical problem that has received wide attention is *influence maximization*: given a social network along with pairwise influence probabilities between peers, and a number k, find k seed nodes such that activating them at start will eventually lead to the largest number of activated nodes in the network in the expected sense. Following the early work of Domingos and Richardson [10] and Kempe et al. [19], there has been a burst of activity in this area (e.g., see [5, 8, 9, 16, 15, 11]). While the majority of previous studies employ propagation models with discrete time, in recent work, continuous time models have been shown to be more accurate at modeling influence propagation phenomena [11, 15, 25]. We refer the reader to the book [7] for a comprehensive survey and a detailed discussion of recent advances in influence maximization.

COSN'14, October 1–2, 2014, Dublin, Ireland.

http://dx.doi.org/10.1145/2660460.2660483 .

As discussed in [19], the propagation models can be classified into *progressive* and *non-progressive* (NP) models. In progressive models, an inactive node can become active, but once active, a node cannot become inactive. Non-progressive models relax this restriction and allow nodes to repeatedly transition between active and inactive states.

Indeed, an overwhelming majority of studies of information propagation have confined themselves to progressive models. For applications such as buying a product, the progressive assumption makes perfect sense: buying a product is not easily reversible in many cases. On the other hand, there are real applications which are not naturally captured by progressive models. For example, consider a user adopting a mobile app. Over time, its appeal may fade and her usage of the app may decline over time. Her interest in the app may be rejuvenated by a friend telling her about a new cool feature being added to the app at which point, she decides to try the app again and may continue using it once again. Alternatively, whenever a new version of the app is released, the user feels tempted to try it again and may, with some probability, decide to continue using it again. As a second example, it is well known that fashion follows cycles. Choices that are in fashion at the moment may fall out of fashion and may again become fashionable in the future, as it has been recognized that social choices follow cyclic trends [29]. As a third example, there are many applications where users may become active and stay in that state for a period of time before deactivating, such as, adopting a feature on a content sharing site where the feature may be the "like" or "favorite" button for a post, filters (sepia, sketch, outline) for photo editing, or "check-in" to a location or a show. Finally, in epidemiology, it is well known that an infected person may recover from a disease but not necessarily acquire lifelong immunity from the disease, thus being susceptible to the disease. In all the above examples, the phenomena in question are subject to spreading via influence. As we will show with experiments on real datasets in this paper, the use of progressive models for capturing such phenomena leads to considerable error. There is a clear need for a non-progressive model for studying these phenomena.

In their seminal paper, Kempe et al. [19] propose a non-progressive model and show that it can be reduced to a progressive model by replicating each node for every timestamp in the time horizon under consideration, and connecting each node to its neighbors in the previous timestamp. They show that this reduction preserves equivalence, which implies all techniques developed for progressive models can in principle be applied to non-progressive models. However, replicating a large network for each timestamp over a large time horizon will clearly make this approach impractical for large social networks containing millions of nodes. Thus, this approach is largely of theoretical interest.

Another related area is *competitive* influence maximization, where competing parties choose seed nodes in order to maximize the adoption of their product or opinion [7]. Non-progressiveness arises naturally from the perspective of any one party involved in the competition. Our focus in this paper is not competition. As illustrated above, there are several example applications where propagation of information or influence happens in a non-progressive manner and it is our goal to model and study them in this paper.

Influence maximization is known to be a computationally hard problem, even over the relatively simpler progressive models. We don't expect influence maximization to be easier over non-progressive models. We face the challenge, *whether we can design approximation algorithms for influence maximization over non-progressive models that scale to large data sets*. To this end, we first propose a discrete time non-progressive model called DNP. It will turn out that DNP, while accurate at modeling non-progressive phenomena, does not lead to a scalable solution for estimating influence spread. To mitigate this, we propose a *continuous time non-progressive model* (CNP), which models the underlying influence propagation as a Markov process. This model can also capture progressive phenomena by appropriately setting the deactivation parameter of the model. We call this variant CNP-Progressive (CP for short). It is interesting to investigate how CP compares with the state-of-the-art progressive continuous time models such as [11, 15].

A second challenge centers on the question, what should the objective be when selecting seeds with respect to non-progressive models. As opposed to maximizing the *number* of active nodes at some time, as done in progressive models, we argue that it is more appropriate to maximize the *expected time during which nodes may have been active*.

Finally, while example applications demonstrating the value of and need for non-progressive models exist, to date, no empirical studies have compared non-progressive models with their progressive counterparts with an aim of calibrating their accuracy for explaining propagation phenomena over real data sets. This is partly exacerbated by the fact that real non-progressive data sets are relatively difficult to obtain. Can we establish the value of non-progressive models using any publicly available data sets?

In this paper, we address all the above challenges. Specifically, we make the following contributions.

- We propose a discrete time non-progressive model and implement it without graph replication (Section 3).
- We propose an efficient continuous time non-progressive model (Section 4).
- We define the objective of influence maximization as choosing seeds so as to maximize the total expected activation time of nodes. We show that the objective function of total expected activation time is both monotone and submodular. This implies the classic greedy seed selection algorithm, combined with our direct approach for computing expected total activation time, provides a $(1 - 1/e)$-approximation to the optimal solution (Section 5).
- Through experiments on synthetic and real datasets, we show that the accuracy of our non-progressive model for estimating expected total activation time is much higher than its progressive counterparts, including the recently proposed continuous time model [11]. Further, we show that our method is more than one order of magnitude faster than an efficient implementation of the DNP model, whose accuracy is comparable to that of CNP. We also show that on progressive data sets, our method using CP is 17-20 times faster than the continuous time progressive model of [11] (Section 6).

We start by presenting related work in Section 2, and conclude with a summary of the paper and a discussion on future work in Section 7. We consciously omit experiments on influence maximization per se, since the outcome of comparisons between the competing methods is obvious. The major bottleneck in scaling influence maximization is in estimating the spread (in our case, expected active time). Our CNP model significantly outperforms the competition on this step and it's trivial to see this advantage will carry over to influence maximization.

2. RELATED WORK

Bharathi et al. [2] use exponential distribution to model the information propagation delay between nodes, and use this to avoid tie-breaking for simultaneous activation attempts by multiple neighbors. We share with them the use of exponential distribution to model activation delays in our CNP model. However, their main goal is designing response strategies to competing cascades rather than maximizing the spread. Considerable work on non-progressive models has been done by the economics community [4]. But they do not focus on computational issues, especially in relation to influence spread computation and maximization.

Kempe et al. [19] proposed several propagation models, including non-progressive ones, but all based on discrete time. Indeed, the DNP model we describe is fashioned after the non-progressive LT model they describe. As we show, our continuous time model CNP significantly outperforms DNP in terms of scalability. Our model and contributions are orthogonal to theirs. In particular, our efficient sampling strategy enables a scalable implementation of influence maximization. Recently, non-progressive models have received attention from the research community [13, 12, 22, 27]. As observed in [13], progressive models are not accurate and there is scalability issue with non-progressive models. Their model is a simplistic model based on strict majority. While theoretically appealing, it's easy to show it's not submodular and no scalable influence maximization algorithm is provided. Furthermore, they focus on finding a perfect target set, one that ends up activating every node, not a realistic goal. Maximizing the overall activation times of nodes is more realistic goal for a business, which is what we study. Other works such as [28, 14, 20, 3] study related problems where nodes have active and inactive states. However, these are significantly different from influence maximization. See [23] for a details.

Finally, a continuous-time Markov chains based progressive model was proposed by Rodriguez et al. [15], and more recently improved upon by Du et al. [11]. However, their methods do not scale well as the time complexity of their solution can be exponentially large for "dense networks", which the authors define as networks with average node degree > 2.5. By that definition, most social networks are dense. Although the authors propose speed-ups that provide approximate solutions or sparsify the networks, their experiments are run on small graphs of at most 1000 edges. In comparison, we evaluate our model on graphs with nearly 30 million edges. Furthermore, it is not easy to directly extend their model to the non-progressive setting. In our experiments, we show that both CNP and its progressive variant CP run 17-20 times faster than [11].

3. DISCRETE TIME NP MODEL

There are two popular influence propagation models: independent cascade (IC) and linear threshold (LT) proposed by Kempe et al. in [19]. They also described an intuitive non-progressive extension of the discrete time LT model. Fundamentally, the models we propose in the next sections are close to the IC model. To set the proper context, in this section, we describe a discrete time non-progressive model that is inspired by the framework given in [19], but closer to the framework we will follow for our CNP model.

Let $G = (V, E, P)$ be a weighted, directed graph representing a social network, with nodes (users) V and edges (social ties) E, with the function $P : E \to [0, 1]$ representing the probability of influence along edges: $P(u, v) := P_{u,v}$ on edge $(u, v) \in E$ is the probability that node v will be activated at time $t + 1$ given that u is active at time t. Additionally, the function $q : V \to [0, 1]$ associates each node $u \in V$ with a deactivation probability: $q(u) := q_u$ represents the probability that u will deactivate at time $t+1$ given that it's active at t. These are the key ingredients of our discrete time non-progressive model. Given the social network graph and a seed set of nodes S that are active at the start of the propagation process, time unfolds in discrete steps. At time $t = 0$, nodes in S are active. At any time $t > 0$, each of the currently active nodes u makes one attempt at activating each of its neighbors v and succeeds with probability $P_{u,v}$. At any time, an active node u can deactivate with probability q_u. We refer to this model as the *discrete-time non-progressive* (DNP) model.

In non-progressive models, nodes can get activated and deactivated for infinitely often, so the influence propagation process can continue indefinitely. Thus, we need to consider an arbitrary but fixed *time horizon* as the time period within which we would like to study the propagation process. Kempe et al. [19] showed that their non-progressive (LT) model's behavior over a given time horizon T can be simulated using a progressive model. The key is to replicate the social network graph for each timestamp. However, a naïve implementation with replicated graphs is not practical. We describe a space efficient implementation that avoids graph replication in our tech report [23]. We show that the DNP model still suffers from a serious inefficiency that each time step, each nodes needs to make the decision of whether or not it changes its state. Thus, at each time step, n nodes need to sample a uniform distribution to determine their state at the next time step. Several nodes may stay in their current state for long periods of time. Hence, sampling at each time step at each node is extremely inefficient. We therefore move to the continuous-time regime for efficiently modeling the non-progressive phenomena.

4. CONTINUOUS-TIME NP MODEL

4.1 Model description

We model influence propagation as a continuous-time Markov process with nodes being in one of two states: *active* and *inactive*. As in classical propagation models, in our model, events trigger state changes and happen probabilistically. We start with a seed set of active nodes. At any time, there are two events that may happen at an active node: the node may activate its neighbor, or may deactivate itself. Similarly, for any inactive node, the node may get activated by one of its active neighbors, or stay inactive. We refer to an event that activates an inactive node as an *activation event* and one that deactivates an already active node as a *deactivation event*. It is these deactivation events that allow the model to be non-progressive.

More specifically, there are two parameters, one for activation and the other for deactivation, both being exponentially distributed random variables. Each edge $(u, v) \in E$ has an associated activation rate parameter $\gamma_{+,u,v}$, and each node u has a deactivation rate parameter $\gamma_{-,u}$. We start with a seed set of nodes that are, by definition, active at time 0. For each node u that is activated at time t, (a) a time τ sampled according to rate parameter $\gamma_{+,u,v}$ has the semantic that v will be activated no later than $t + \tau$, and (b) a time τ' sampled according to rate parameter $\gamma_{-,u}$, has the semantic that node u will deactivate at time $t + \tau'$. Notice that another neighbor of v may activate it sooner. In particular, an inactive node v that is reachable from one or more active nodes activates at a time equal to the shortest path from those active nodes, that is shortest in terms of the sum of sampled propagation times of the edges forming the path. However, each activation or deactivation with its associated rate parameter is one *local* event. That is, only the ego-centric network of a node is involved in any event. This observation is key to the scalability of our proposed in terms of implementation. In particular, unlike the recently continuous time (but progressive) models [11, 15], we don't need to compute or even estimate the shortest path length directly. In Section **??** we show how these parameters can be learned from data. Notice that unlike the discrete time counterpart, the model parameters govern the times at which events happen as opposed to whether the events will happen. This is a direct consequence of moving to continuous time.

4.2 Semantics of the propagation

During an influence propagation cascade, there are multiple activation and deactivation events that may happen. In order to model the cascade, we need to find the one that happens first and update the activation status of the corresponding node. For instance, if u is active, it deactivates with some rate parameter, however, it is also trying to activate its inactive neighbor v with some rate parameter. If u deactivates before activating v, then v may not have a chance to activate (assuming it has only one neighbor) unless u activates again. Further, if there are multiple neighbors trying to activate a node v, it will get activated by the local event that happens first, i.e., by the neighbor that first activates it. Therefore, it is important to understand and model the order of events. We crucially make use of two key properties of exponential distributions for modeling the time and order of events.

PROPERTY 1. *For n different events with rate parameters $\gamma_1, \gamma_2 \ldots \gamma_n$, the probability that the i^{th} event will happen first is $\frac{\gamma_i}{\sum_{i=1}^{n} \gamma_i}$.*

PROPERTY 2. *For different events with rate parameters $\gamma_1, \gamma_2 \ldots \gamma_n$, the time of the first event is exponentially distributed with rate parameter: $\sum_{i=1}^{n} \gamma_i$.*

We keep track of the current time, t_{cur} during a propagation process. At each iteration, the categorical distribution in Property 1 is sampled to determine the event that happens first (or next). Then, the exponential distribution with rate parameter $\sum_{i=1}^{n} \gamma_i$ is sampled (Property 2) to obtain the time elapsed τ between last event and this event. The current time is then updated as $t_{cur} = t_{cur} + \tau$, and we proceed to the next iteration if $t_{cur} < T$, where T is the time horizon, and stop otherwise. In other words, even though the model is continuous time, it has a clear interpretation in terms of discrete steps, namely the occurrence of events.

Another way to understand the model semantics is in terms of possible worlds. A deterministic possible world for our model can be constructed as follows: For each edge $(u, v) \in E$ we sample an array of timestamps and sort it. A timestamp in the array indicates that if u is active at that time, it will activate node v. We call this array the *schedule of activations*. Similarly, for each node $u \in V$, we sample an array of deactivation times. If u is active at those timestamps, it will get deactivated. We refer to this array as the *schedule of deactivations*. The set of possible worlds for a given instance of our CNP model is the set of all such edge activation schedules and node deactivation schedules, for every edge and node in the given social graph. Such a construction of possible world aptly covers all possibilities in our random process. We will use these semantics to prove monotonicity and submodularity of the spread under the CNP model in Section 5.

4.3 Advantages of CNP over DNP

If we correctly map the rate parameters in CNP model to the probabilities in DNP model, the simulation results of two models will be similar. We note that the models are not equivalent, but have similar accuracy in terms of the expected spread, when the following mapping holds. In CNP, for any edge (u, v) where u is active but v is not, the probability that u activates v within the next time unit is equal to the CDF$(1, \gamma)$, where CDF is the cumulative distribution function of the exponential distribution, γ is the rate parameter associated with (u, v), and 1 is the time unit. The corresponding edge probabilities in the DNP model would be CDF$(1, \gamma)$. Similarly, we map the deactivation rates in CNP to deactivation probabilities in DNP. Then, the resulting DNP model will be a discrete-time approximation of the CNP model. Therefore, we expect the accuracy of CNP and DNP to be similar.

We now compare the two models in terms of the computational cost incurred at each activation and deactivation. In the discrete time case, for each active node, we need to sample from a uniform distribution once at each timestamp to determine whether or not the node deactivates. In the continuous-time setting, however, we first need to randomly choose the event that occurs with probability governed by Property 1, then we need to sample the exponential distribution to get the time at which it occurs, using Property 2. Therefore, for nodes that do not deactivate in the time window, their cost of (attempted) deactivation is zero in the continuous-time setting, again, a significant saving from the discrete-time regime.

5. INFLUENCE MAXIMIZATION

Next, we discuss influence maximization, i.e., the process of seed selection to maximize the spread of influence under the CNP model. The influence maximization problem for non-progressive models is similar to that described in [19]. However, since nodes can deactivate, the *spread*, traditionally defined as the expected number of active nodes, changes with time. Thus, maximizing the expected number of active nodes at a given timestamp, or at the time horizon may not be ideal from the point of view of company initiating a viral marketing campaign. We start by proposing an intuitive objective function for spread under a non-progressive model. Importantly, we show that our proposed spread function is monotone and submodular, hence the greedy approach to maximize the function can be applied.

Objective Function. In a non-progressive world, an intuitive objective from the point of view of a marketeer is to maximize the "active time" of its customers in a given social network. That is, maximize the total amount of time that nodes in the network are active, in expectation. Given a seed set A,

$$spread_A = \sum_{v \in V} \tau_v$$

where τ_v is the sum of time intervals for which node v is active. Then, the influence maximization problem [19] is defined as: select a seed set of nodes $A \subseteq V$ to be activated such that the expected $spread_A$ is maximized over a chosen time horizon T, given the non-progressive influence propagation model.

Monotonicity and Submodularity. As an important step towards solving the influence maximization problem, we show that the expected *spread* is monotone and submodular. Then, we can use the state-of-the-art greedy algorithm, such as CELF [21] and CELF++ [17], to guarantee a $1 - 1/\epsilon$ approximation. It is easy to see that,

$$\mathrm{E}[spread_A] = \sum_{v \in V} \mathrm{E}[\tau_v] = \int_{t=0}^{T} \mathrm{E}[\sigma(A, t)] dt$$

where $\sigma(A, t) = |S|$, S is the set of nodes activated from the seed set A at timestamp t, and $\sigma(A, t)$ is the number such nodes or the cardinality of set S. Therefore, we can prove monotonicity and submodularity of the expected spread, by showing that these properties hold for $\mathrm{E}[\sigma(A, t)]$. For this, we follow the proof guidelines in [19] to construct a deterministic possible world from the random process that we are modeling. Let X be the set of all possible worlds, and given $x \in X$, let $pdf(x)$ denote the probability density function of x. Then,

$$\mathrm{E}[\sigma(A, t)] = \int_{x \in X} pdf(x) \times \sigma_x(A, t) dx$$

Thus, we only need to prove that $\sigma_x(A, t)$ is monotone and submodular. Note, that we need to integrate over the possible worlds, as opposed to a summation performed in [19], because the number of deterministic possible worlds is uncountable in our setting.

LEMMA 1. *Additivity of spreads: Given two sets of seed nodes A, B, timestamp t, and a possible world x,*

$$S_x(A \cup B, t) = S_x(A, t) \cup S_x(B, t)$$

where $S_x(A, t)$ denotes the set of nodes activated by seed set A in possible world x at timestamp t.

THEOREM 1. *Given lemma 1, $\sigma_x(A, t) = |S_x(A, t)|$ is monotone and submodular.*

See proofs in our tech report [23].

6. EXPERIMENTAL EVALUATION

In this section we compare the accuracy and running time of: traditional IC model, state-of-the-art continuous time progressive model ConTinEst[11], DNP and CNP, for estimating the spread as defined in Section 5. We evaluate our model on synthetically generated data and two real datasets: Flixster and Flickr, for which we have a social network, and an action log which contains the timestamps of users' actions. The synthetically generated dataset consists of a 500 node graph, and randomly generated cascades and deactivation events. The Flixster dataset [18] has 1 million nodes (users) and 26.7 million edges (social connections). An activation event is the act of *rating a movie from a specific genre*. Finally, the Flickr dataset [6] has 2.3 million nodes and 33.1 million edges. An action corresponds to using the "favorite photo" feature. We provide a detailed description of how our model parameters can be learned from data, and our experimental setup in our tech report [23]. For implementing the sampling without replacement procedure for a categorical distribution, see methods described in [24].

6.1 Comparison Across Models

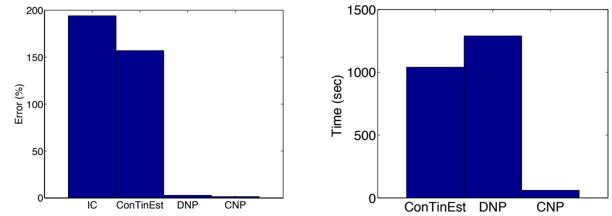
We start with presenting an overview of our results comparing different models: progressive vs. non-progressive models, and discrete vs. continuous time models, across two axes: accuracy of spread estimation and running time. Figure 1 illustrates this comparison for Flixster dataset. See detailed numbers for both datasets in Table 1. We make the following key observations, and substantiate these with details through the remainder of this section.

- Progressive models, here classical IC model and state-of-the-art ConTinEst, overestimate the spread and result in an error of 80-194% compared with the ground truth.

- Non-progressive models, DNP and CNP, are highly accurate in estimating the spread with very small errors of 0.1-3%. Notice that DNP is the non-progressive counterpart of the progressive IC model model and improves the accuracy or error in estimating spread by a 100%.

- Continuous models ConTinEst and CNP are slightly better at estimating accuracy than the discrete models IC and DNP.

- Our model CNP is not just more accurate but also an order of magnitude faster than the state-of-the-art continuous time model ConTinEst (Figure 1(b)). These results are for running 100 Monte-Carlo simulations.



(a) Error in Spread estimation (b) Running time

Figure 1: Spread estimation error and time comparison for progressive (IC, ConTinEst) and non-progressive models (DNP, CNP) for Flixster dataset

Dataset	Ground Truth	CNP	DNP	ConTinEst [11]	IC
Flixster	964013	949750 (1.5%)	991141 (2.8%)	2477678 (157%)	2833860 (194%)
Flickr	2435663	2432053 (0.148%)	2409695 (1.07%)	4423372 (81%)	4922860 (102%)

Table 1: Spread estimated by progressive (IC, ConTinEst) and non-progressive models (DNP, CNP) and error percentage w.r.t. ground truth

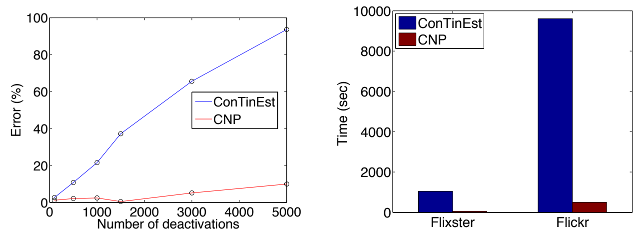
Evaluating Accuracy. We evaluate the accuracy of the estimation of spread of our model by simulating the propagation, starting at the state of the network at the last timestamp in our training set, and evaluating against the ground truth of spread achieved in the test set. In other words, the nodes active at the end of the training set are treated as the seed set, and the propagation is run for the time horizon equal to the length of the test set. The *ground truth* of spread is computed as the total active time of all nodes for the test set. Table 1 shows the spread as estimated by our model compared with the ground truth. For our model, error in spread estimation is just 1.5% and 0.1% over the Flixster and Flickr datasets resp. The difference between IC model and non-progressive models is two orders of magnitude. This validates that deactivation occurs in real datasets, and that modeling deactivation properly is critical for a reasonable estimation of influence spread in non-progressive settings.

Next, we perform an experiment on synthetic data to show the impact of number of deactivations on the spread estimates by a progressive model (ConTinEst [11]) and our CNP model. Figure 2(a) shows this the error (%) in estimating the spread. As the deactivations increase, the gap in the accuracy of the two methods increase, with CNP performing over 83% better than the competitor, establishing that in the presence of deactivations, non-progressive phenomena are modeled accurately by CNP.

Evaluating Computational Cost. We compare the running time of ConTinEst with CNP for our two real datasets Flickr and Flixster for running 100 Monte-Carlo simulations. As seen in Figure 2(b), our model is an order of magnitude faster than its progressive competitor.

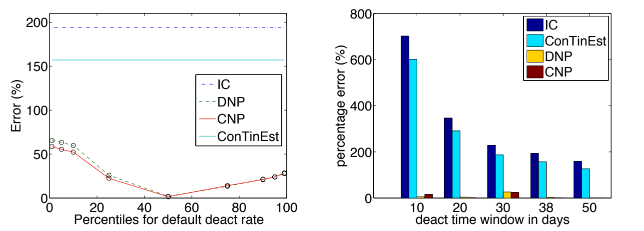
6.2 Varying Parameter Values

Effect of deactivation parameters on accuracy. For Flickr, we observe that 98% of the nodes perform no action in our training set, and hence get a zero deactivation rate. This is an artifact of the short timespan of the training data.



(a) Error in Spread estimation for synthetic data (b) Running time of ConTinEst vs. CNP on real data

Figure 2: Accuracy comparison for progressive (IC, ConTinEst) and non-progressive models (DNP, CNP)



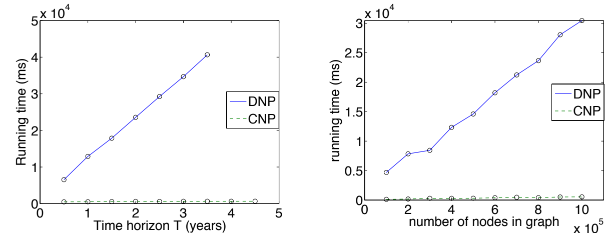
(a) Varying deactivation rate (b) Varying deactivation time window for Flixster

Figure 3: Accuracy comparison for progressive (IC, ConTinEst) and non-progressive models (DNP, CNP) with varying deactivation parameters

Filtering those nodes would result in a disconnected graph. To overcome this shortcoming of the data sample and to avoid overfitting, we assign a *default deactivation rate* to all such nodes. We use the set of non-zero deactivation rates (as learned from the data) as a guideline, and test different percentiles of this set as the default deactivation rate. Instead of fixing the value, we evaluate its impact on the accuracy by varying it, as shown in Figure 3(a) for Flixster dataset. As seen in the figure, the default deactivation rate does impact the accuracy slightly, still the estimates by CNP are orders of magnitude more accurate than IC model and ConTinEst. Also notice that the estimated spread for DNP and CNP is very similar validating our argument in Section 4.3 that DNP is an approximation of CNP. The plots for Flickr are skipped for brevity, but the methodology adopted and results were similar. For the remainder of the experiments we set the default deactivation rate for Flixster and Flickr to the best obtained, i.e., 50^{th} and 1 percentile resp. of the unique non-zero deactivation rates learned.

Next, we show using Figure 3(b) that changing the deactivation time window does not significantly impact the accuracy of CNP model. Although the progressive models are unaffected by the deactivation window, the ground truth computed is different across windows, and this change is reflected in the error percentage.

Effect of varying parameters on running time. The two factors that affect the computational cost of simulating the non-progressive models are: time horizon and graph size. We compare the computational cost for CNP and DNP for increasing time horizon on the full graphs of the two datasets. As seen in Figure 4(a), the running time for DNP increases linearly over increasing time horizon, while that



(a) Increasing time horizon (b) Increasing graph size

Figure 4: Running time comparisons on Flixster dataset for non-progressive models: DNP and CNP

for CNP changes only slightly. For instance, for Flickr, the running time for CNP is 73% less than DNP at 27 month time horizon. Finally, we set time horizon as 2 years for Flixster and show the running time with increasing graph size in Figure 4(b). Again, CNP scales up very well. The results for Flickr were similar and skipped for brevity.

Progressive Setting. We ask the question, "What if the world is progressive, i.e., there are no deactivations, how well would CNP perform?" To this end we perform an experiment of setting the deactivation time window to the end of the time horizon, essentially saying no nodes deactivate. We then compare this model we call CP for the continuous progressive version of our proposed model against ConTinEst and CNP. We observe that the running time on Flixster for CP, CNP and ConTinEst are 60, 64 and 1041s resp. The running time results for Flickr for the CP, CNP and ConTinEst were 498, 606 and 9605s resp. This illustrates that despite the data being progressive in nature, our model is 17-20 times faster than the state-of-the-art progressive continuous model.

7. CONCLUSIONS

There are applications where the propagation phenomena are more accurately captured using non-progressive models. In their seminal paper, Kempe et al. [19] proposed a non-progressive LT model and showed that over any finite time horizon of interest, its behavior can be effectively simulated by a progressive model with the given social graph replicated at every timestamp in the horizon. Inspired by this, we proposed a non-progressive model and showed that its behavior over a time horizon can be simulated without any need for graph replication. The resulting discrete time non-progressive model is still not scalable owing to the prohibitive number of samplings necessary in order to monitor the state of nodes at every time. We proposed an alternative continuous time non-progressive model and showed that it permits a highly efficient implementation. In place of expected number of active nodes, for our continuous time model, we motivated the expected total amount of time the nodes in the network are active, as the right notion of spread, which a seed selection algorithm should optimize. We showed that this objective function is monotone and submodular in the set of seed nodes. By extensive experiments on two data sets, we show that our model significantly outperforms the progressive state of the art, ConTinEst [11] both on accuracy of spread estimating and on running time. It would be interesting to study non-progressive continuous time models in the competitive setting, where advertisers may be adversarial towards competitors.

8. REFERENCES

[1] J. G. U. A. Ostfeld and E. Salomons. Battle of water sensor networks: A design challenge for engineers and algorithms. In *WSDA*, 2006.

[2] S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. In *WINE*, 2007.

[3] G.-I. Bischi and U. Merlone. Global dynamics in adaptive models of collective choice with social influence. In *Mathematical modeling of collective behavior in socio-economic and life sciences*, pages 223–244. Springer, 2010.

[4] L. Blume. The statistical mechanics of strategic interaction. *Games and Economic behavior*, 1993.

[5] C. Budak, D. Agrawal, and A. E. Abbadi. Limiting the spread of misinformation in social networks. In *WWW'11*.

[6] M. Cha, A. Mislove, and P. K. Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *WWW*, 2009.

[7] W. Chen, L. V. S. Lakshmanan, and C. Castillo. *Information and Influence Propagation in Social Networks*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2013.

[8] W. Chen, Y. Wang, and S. Yang. Effcient influence maximization in social networks. In *KDD*, 2009.

[9] W. Chen, Y. Wang, and L. Zhang. Scalable influence maximization in social networks under linear threshold model. In *ICDM*, 2010.

[10] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2001.

[11] N. Du, L. Song, M. Gomez-rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*. 2013.

[12] E. Even-Dar and A. Shapira. A note on maximizing the spread of influence in social networks. In *Internet and Network Economics*, volume 4858. Springer Berlin Heidelberg, 2007.

[13] M. Fazli, M. Godsi, J. Habibi, P. J. Khalilabadi, V. Mirrokni, and S. S. Sadeghabad. On the non-progressive spread of influence through social networks. In *LATIN'12*.

[14] A. Ganesh, L. Massoulié, and D. Towsley. The effect of network topology on the spread of epidemics. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 1455–1466. IEEE, 2005.

[15] M. Gomez-Rodriguez and B. Scholkopf. Influence maximization in continuous time diffusion networks. In *ICML*, 2012.

[16] A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[17] A. Goyal, W. Lu, and L. V. Lakshmanan. Celf++:optimizing the greedy algorithm for influence maximization in social networks. In *WWW*, 2011.

[18] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys*, 2010.

[19] D. Kempe, J. Kleinbery, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[20] C. Kuhlman, V. Kumar, M. Marathe, S. Swarup, G. Tuli, S. Ravi, and D. Rosenkrantz. Inhibiting the diffusion of contagions in bi-threshold systems: Analytical and experimental results. In *Proceedings of the AAAI Fall 2011 Symposium on Complex Adaptive Systems (CAS-AAAI 2011)*, pages 91–100, 2011.

[21] J. Leskovec, A. Kraus, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. S. Glance. Cost-efficiective outbreak detection in networks. In *KDD*, 2007.

[22] Y. Li, W. Chen, Y. Wang, and Z.-L. Zhang. Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. In *WSDM*, 2013.

[23] V. Lou, S. Bhagat, L. V. S. Lakshmanan, and S. Vaswani. Modeling non-progressive phenomena for influence propagation, 2014. `http://arxiv.org/abs/`.

[24] Y. Matias, J. S. Vitter, and W.-C. Ni. Dynamic generation of discrete random variates. In *SODA*, pages 361–370, 1993.

[25] M. F. B. Nan Du, Yingyu Liang and L. Song. Continuous-time influence maximization for multiple items. *CoRR*, abs/1312.2164, 2013.

[26] A. Ostfeld and E. Salomons. Optimal layout of early warning detection stations for water distribution systems security. In *J. Water Resources Planning and Management*, 2004.

[27] N. Pathak, A. Banerjee, and J. Srivastava. A generalized linear threshold model for multiple cascades. In *ICDM*, 2010.

[28] B. A. Prakash, D. Chakrabarti, N. C. Valler, M. Faloutsos, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. *Knowledge and information systems*, 33(3):549–575, 2012.

[29] A. D. Sarma, S. Gollapudi, R. Panigrahy, and L. Zhang. Understanding cyclic trends in social choices. In *WSDM*, pages 593–602, 2012.

[30] R. Schenkel, T. Crecelius, M. Kacimi, S. Michel, T. Neumann, J. X. Parreira, and G. Weikum. Efficient top-k querying over social-tagging networks. In *SIGIR'08*.

[31] T. Valente. Network models and methods for studying the diffusion of innovations. In *Models and methods in social network analysis*. Cambridge Univ. Press, 2005.