Content Recommendation for Viral Social Influence

ABSTRACT

How do we create content that will become viral in a whole network after we share it with friends or followers? Significant research activity has been dedicated to the problem of strategically selecting a *seed set* of initial adopters so as to maximize a meme's spread in a network. Yet this line of work assumes that the success of such a campaign depends solely on the choice of a *tunable* seed set of adopters, regardless of how users perceive the propagated meme, which is *fixed*. Yet in many real-world settings, the opposite holds: a meme's propagation depends on users' perceptions of its *tunable* characteristics, while the set of initiators is *fixed*.

In this paper, we address the natural problem that arises in such circumstances: Suggest content, expressed as a limited set of *attributes*, for a creative promotion campaign that starts out from a given seed set of initiators, so as to maximize its expected spread over a social network. To our knowledge, no previous work addresses this problem. We find that the problem is NP-hard and inapproximable. As a tight approximation guarantee is not admissible, we design an efficient heuristic, Explore-Update, as well as a conventional Greedy solution. Our experimental evaluation demonstrates that Explore-Update selects near-optimal attribute sets with real data, achieves 30% higher spread than baselines, and runs an order of magnitude faster than the Greedy solution.

ACM Reference format:

. 2016. Content Recommendation for Viral Social Influence. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 10 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Online networking offers opportunities for new types of marketing. A prime example of such a new marketing technique is viral marketing, whereby organizations run promotion campaigns through word-of-mouth effects within online social networks. The *influence maximization* (IM) problem [27], studied intensively during the last decade, aims to find well-chosen *seed nodes* from which to launch such campaigns so as to achieve good results.

Recent works [17, 19] have focused on the parameters that define the popularity of a post, campaign, idea, or *meme* within a network. Such works were the first to study the question of how commercial brand posts engage online social network users, drawing from the theory of Uses & Gratifications [26]; they examine post parameters such as *content type* (e.g., entertaining, informational), *media type* (e.g., vivid, interactive), *posting time* (e.g., workday, peak hours) and *valence of comments* (e.g., positive, negative). Interestingly, such studies have reached some ambivalent conclusions; for instance, [19] ascertains that entertaining content decreases the number of likes, while [17] claims the exact opposite.

Concurrent research has studied the problem of *viral product design* [3, 6], which calls for engineering products by incorporating viral attributes so as to generate peer-to-peer influence that encourages adoption within a network. Aral and Walker [3] study the question of viral attribute selection under randomized trials; Barbieri and Bonchi [6] study the same problem as a complement to the standard IM problem of selecting a set of seed nodes that maximizes influence. Both these works conceive of viral attributes as attached to products themselves; that is, they do not investigate the more general problem of choosing content, out of a large set of eligible content options, for any kind of *meme* spreading in a network, so as to make it viral. Other works study the problem of selecting tuple attributes or generating snippets so as to maximize visibility [18, 31]; these works do not pertain to a social network setting.

In this paper, we introduce and study the problem of selecting content that characterizes any type of *meme*, so as to maximize its expected spread through a network, starting out from a fixed set of initial adopters. For instance, an advertisement post may feature aspects such as *topics*, *people*, *locations* and *abstract themes*. We are particularly interested in those content aspects that are associated with specific *online social network pages*; we denote such aspects as **content attributes**. Fittingly, online social network users themselves are associated with such non-personal network pages: they express their preferences for specific brands, topics of interest, public persons, hobbies, or locations by subscribing to or "liking" such pages. Thereby, an attribute's popularity can be gauged via its number of subscribers or page "likes". For our purposes, we denote the pages that a user subscribes to or "likes" as **user attributes**. We contend that, the more content and user attributes overlap, the more likely that user is to propagate that post. We envisage an organization that aims to achieve high viral effect of a campaign initiated from its fixed set of subscribers. For example, assume *FlyFast* airways wants to launch a promotion campaign in social media. *FlyFast* already has a social network presence, and its page has a subscribers' set S fixed at a given moment, while it faces constraints related to its budget and people's attention span. In their design, *FlyFast* consultants are interested to identify a set of k content attributes, out of a universe of eligible, mutually compatible options, that will maximize the expected network spread of a post starting out from its subscribers' set S. Assume that, for $k = 4$, the optimal attribute set is {*"Best travel Accessories"*, *"Airline food guide"*, *"Hipster Europe"*, *"Backpacker tips"*}. Guided by this knowledge, *FlyFast* can infuse its post with complementary content that appeals to users interested in those topics, e.g., promotions to backpackers, references to its hipster audience, and highlights on its food quality. Thereby, it can maximize its promotion's reach.

To our knowledge, we are the first to study the influence maximization problem in which the seed is *given* and post content is *sought*. Our related contributions are as follows:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, Washington, DC, USA
© 2016 ACM. 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

Problem Setting We motivate the influence maximization problem in settings where the set of initial adopters is fixed, or even a single point of origin, and the content of a propagated meme can be tuned. We formulate the concept of *digital influence* as a special case of social influence.

Propagation Model We devise a *content-aware propagation model*, whereby the probability of influence across edges depends on content attributes. We show that, with this model: (i) the problem of choosing content attributes that maximize influence is NP-hard; (ii) the spread function is not submodular, hence no submodularity-based approximation algorithm applies; and (iii) it is NP-hard to approximate the optimal solution within a factor of $n^{1-\varepsilon}$ for $\varepsilon > 0$.

Algorithm We design a fast algorithm, *Explore-Update*, which achieves higher influence spread than baselines; its effectiveness is based on the iterative estimation of the marginal spread achieved by each attribute, while its efficiency is gained by limiting such computations only to nodes within a probability-based distance threshold θ and attributes potentially affecting such nodes.

Experiments We compare Explore-Update to two baselines and show that it always achieves better propagation results, while it is significantly faster than a naïve Greedy approach; we calculate the optimal solution on a reduced dataset with a small universe of attributes, showing Explore-Update can achieve optimality; last, we demonstrate the scalability of Explore-Update on seed set size.

2 MOTIVATION AND BACKGROUND

We start our discussion drawing from cognitive science and marketing, paving the way for our problem definition.

2.1 Idea Habitats

An *idea habitat* [8] is the set of environmental cues that make people think about an idea and pass it along. Regardless of how well an idea is encoded, it will persist and spread only if the environment cues people to retrieve it regularly. In other words, even if an idea *can* be easily recalled, if it is only rarely cued by the environment, it may remain rare and be forgotten. It follows that, for an idea to spread, it should be not only well encoded and easily recalled, but also regularly retrieved. Promotion campaigns aim to assist the spread of any meme in the same way as idea habitats assist the spread of ideas. For instance, assume that *FlyFast* food is good and therefore memorable. Still, without a viral promotion, *FlyFast* will miss the spotlight, letting other airlines gain public attention.

2.2 Digital Influence

Social influence is defined in terms of peer influence, based on peer behavior [1], so as to distinguish it from confounding factors such as simultaneity [22], unobserved heterogeneity [20], homophily [2] and reflection [30]:

Definition 1. ***Social Influence*** *expresses the extent to which the behavior of one's peers changes the utility one expects to receive from engaging in a certain behavior, and hence the likelihood that one will engage in that behavior.*

This definition can be used to make an argument that, if we understand how behaviors spread from person to person, our society will be able to promote agreeable behaviors, such as physical exercise and financial responsibility, and limit disagreeable ones, such as violence and dirty needle sharing. By this definition, peer behaviors may relate to awareness, persuasiveness, imitation and social learning [1]. We propose that *digital influence* can be seen as a case of awareness-related social influence, defined as follows:

Definition 2. ***Digital Influence*** *expresses the extent to which the content of a commercial digital posting changes the support one wants to provide to that posting, and thus the likelihood that one will propagate it.*

In online social networks, posts are propagated from user to user by means of actions such as *like*, *share*, or *repost* [5]. Without loss of generality, we group all these actions under the *like* action. What matters is whether users endorse and promote a post further by making it visible to their friends and followers. In our setting, we emphasize the importance of earning *likes* from users at large.

Nowadays, as online social network users are exposed to a plethora of posts, we can safely assume that they become selective on what they *like*. Therefore, a brand that fails to issue likable posts via its social network pages is unlikely to spread awareness of its activities and products. In the same vein, a brand that can estimate how viral a post will be and create appropriate digital content, stands good chances to succeed. We argue that such estimation can be based on the digital influence of content associated with a meme; this observation brings us to the topic of *influence maximization.*

2.3 Influence Maximization

The classic Influence Maximization (IM) problem, formulated by Kempe et al. [27], has been intensively studied over the last decade. Recently, the focus has shifted to providing realistic definitions to the concept of *influence spread.* Barbieri et al. [7] proposed the *Topic-Aware Influence Cascade* (TIC) and *Topic-Aware Linear Threshold* (TLT) models, which are extensions of the IC and LT models [27].

Influence Maximization in Social Networks. The first solutions to the IM problem were proposed by Domingos and Richardson [21, 33], yet had no guarantees on influence spread. Then, Kempe et al. [27] formulated the problem based on the Independent Cascade and Linear Threshold propagation models, proved its NP-hardness, and proposed a greedy algorithm with a $(1 - 1/e - \epsilon)$ approximation guarantee. Subsequent works investigated efficiency and scalability questions, either with heuristics [13, 14, 16] or preserving an approximation guarantee [9, 15, 23, 28, 34].

Topic-Aware Influence Maximization. Barbieri et al. [7] were the first to look at social influence taking content characteristics into consideration. They proposed methods that learn propagation model parameters such as topic-aware influence strength from a query log of past propagation traces, and verified experimentally that a larger influence spread can be engendered when taking item characteristics into consideration via their Topic-Aware Influence Maximization (TIM) models.

Aslay et al. [4] studied online TIM queries; the incentive for this online scenario is that many independent advertisers wish to instantly detect the k most influential users for advertising purposes; each advertisement contains a different set of keywords and hence induces a new probabilistic graph creating a separate TIM instance; the authors proposed an offline-online solution, INFLEX, based on an index used to identify similarities among a new and log

TIM queries; pre-computed solutions for log queries are aggregated online so as to provide an approximate solution for a TIM query.

The online TIM problem is also studied in [10, 12]. Chen et al. [12] investigated the topic-aware influence results of two real networks and utilized the derived properties to form three preprocessing-based algorithms, of which MIS is the best; its main difference from INFLEX is that, in MIS, pre-computed seed sets are based on each separate topic and not on a mixture of topics from different log queries. Chen et al. [10] utilized the maximum influence arborescence (MIA) model [13] to achieve high influence spread with a theoretical guarantee. The core idea is to utilize upper- and lower-bounding techniques, so that the exact marginal influence is computed only for the most promising nodes. This work provides the state-of-the-art solution for the online TIM problem [4].

Recently, Li et al. [29] investigated the online TIM problem differently, proposing the alternative problem of Keyword-Based Targeted Influence Maximization (KB-TIM). By KB-TIM, each user is represented as a weighted vector of preferences for different keywords, which represent topics. This vector can be generated by applying topic modeling techniques [24] on aggregated user social activities, such as posts, likes, etc. An advertisement then achieves an impact determined by its own topic-oriented keywords. The KB-TIM problem aims to maximize an advertisement's impact, considering only influence on target users relevant to that advertisement. The solution in [29] is motivated by previous work in [34]. The main difference is that, while in [34] θ users for a sampled Reverse Reachable (RR) set [9] are randomly selected, in [29] those θ selected users are the ones on which a high advertisement impact can be exerted. Moreover, two disk-based indexing methods are proposed, that precompute the sampled RR sets for different keywords so as to utilize online those RR sets associated with query keywords. Yet results in [29] are not compared to those in [10].

Our work differs drastically from all aforementioned works: rather than aiming to detect an influential set of users given a post's content attributes, we are interested to find viral content attributes themselves, given a set of initial adopting users, so as to form an influential post. The aforementioned works do not examine what kind of posts would be most promising or powerful given all topics in the network. Besides, the topic-aware approach is based on general topical terms, like music, soccer, cars, etc., ignoring the high variation among different specimens within such terms. In contrast, we search for specific attributes that can form posts successful in terms of influence spread.

Influence Maximization with VPD. Aral and Walker [3] investigated the problem of viral product design under randomized trials focusing on product features like personalized referrals and broadcast notifications. Thereafter, Barbieri and Bonchi [6] studied the problem of influence maximization *in conjunction with* that of viral product design, aiming to detect a combination of seed nodes and product attributes that maximize influence in a network. The proposed solutions are generic methods named *Local Update* and *Genetic Update*; the former is a greedy algorithm allowing for both addition and removal of attributes at each greedy iteration; the latter is a brute-force method that randomly selects a subset of all attributes. By contrast, we examine attribute selection for a post (not a product) as a stand-alone problem in its own right.

2.4 Distinctiveness

We emphasize three distinct elements of this work.

Problem Formulation While our model takes into consideration the influence with regard to a post exercised by users themselves, we seek to maximize the influence exercised by the appeal and quality of a post's content within a network. To the best of our knowledge, we are the first to define this problem, which is distinct and cannot be treated by methods aiming at user selection.

Nature of Attributes In related works, attributes are configured as *general topics*; for example, Barbieri and Bonchi [6] assign *tags*, conceived of as general topics, to the role of product attributes. By contrast, in our problem formulation and in our experimental study, an attribute corresponds to a topic of interest *identifiable via a non-personal social network page*. Thus, even a page on an abstract theme (e.g., *Psychology of Relations*) is a possible attribute in our setting: it has a specific commercial value (4,719,837 followers) that differentiates it from other pages on similar topics.

Use of Tags The type of content attributes we investigate can hardly be articulated via tagging. Tags express highly idiosyncratic user impressions, focusing on arbitrary aspects of content; they are volatile as descriptors of content, whereas we need a stable ground truth to represent content. For example, a video post may relate to several specific topics of interest, yet it would be hard to identify these via user tagging. In consequence, past tagging does not offer valuable information in our problem setting.

3 PROBLEM STATEMENT

From the preceding discussion, we conclude that any brand would gain by maximizing the expected effectiveness of its product promotion campaigns within an online social network. We assume that there exists a certain set of *subscribers* to the brand's social network page, and a promotion campaign aims to influence the maximum number of *non-subscribers*; as we discussed, such users are associated with topics expressing their interests.

3.1 Content-Aware Cascade Model

We model an online social network as a directed graph $G = (V, E)$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of nodes, each of which corresponds to an individual user, and $E \subset V \times V$ is a set of directed edges representing social relations among users. Each node v has a set of associated attributes $F_v = \{f_v^1, f_v^2, \ldots\}$, from a universe Φ, that define user preferences; we identify these attributes as the non-personal network pages a user expresses interest in. A posting propagated through the network is associated with a set of attributes $F = \{f_{p_1}, f_{p_2}, \ldots, f_{p_K}\} \subseteq \Phi$, which affect the probability of its propagation across a network edge e_{uv}, along with the attributes F_v associated with the targeted node v.

The *Content-Aware Cascade Model* (AACM) is a variant of the Independent Cascade model (IC), in which edge propagation probabilities depend on the attributes associated with users and with the propagated post. An AACM diffusion process unfolds in steps, starting from an initial seed set of activated nodes. A node u activated at time step t has a single chance to activate its out-neighbors. The process is incremental, as nodes can alternate only from inactive to active states; the diffusion ends when there is no newly activated nodes at a given step. At any step, a newly activated node u activates its out-neighbor v with probability $p(u, v)$ equal to:

$$\begin{aligned} p_{uv} &= b_{uv} + q_{uv} \cdot h_{uv}(F_v, F), \quad b_{uv}, q_{uv} \in [0, 1] \\ h_{uv}(F_v, F) &= \min\left\{\tfrac{1-b_{uv}}{q_{uv}}, |F_v \cap F|\right\} \end{aligned} \tag{1}$$

where b_{uv} is a *base probability* on an edge and q_{uv} a *marginal probability* that indicates how much the probability on an edge increases for each selected attribute in F matching a preference of node v, as indicated by the transition function $h_{uv}(F_v, F)$, with a sanity bound of $\frac{1-b_{uv}}{q_{uv}}$. We emphasize that the marginal probability q_{uv} distinguishes among different user links, albeit not among different attributes for a given link; a more complex model could distinguish among different attributes, or even define a probability distribution function over the set of all attributes [7], to be learned by historical logs. We choose to relegate the problem of defining and learning such probability distribution functions to future work, and now study the problem under the modeling assumption that each attribute has the same independent effect on the probability function. Nevertheless, we emphasize that our simplified model forms a special case of any more complex model in which each attribute would have a different effect on the probability function, i.e., the special case, such effects are rendered equal. Therefore, our subsequent hardness and inapproximability results hold for any such more complex model as well. Furthermore, parameters q_{uv} and b_{uv} can be obtained from past data, as in [7]; in our setting, we assume that such parameters have been obtained in advance.

Given a seed set S of subscribers, for every set of attributes F, we can obtain the total number of activated nodes after running several trials of the diffusion process from S [27]. The *expected* number of activated nodes for a given seed set S and a selected set of attributes F is called *influence spread*, denoted as $\sigma(F|S)$ or, if it is clear from the context, just $\sigma(F)$. Thus, the influence spread $\sigma(F)$ is the *expected* spread of the diffusion process, which we can calculate using live-edge instances of the graph as:

$$\sigma(F) = \sum_{X} Prob[X] \cdot \sigma_X(F) \tag{2}$$

where $\sigma_X(F)$ is the influence spread in live-edge instance X.

3.2 Attributed Post Influence Maximization

We define the APIM problem as follows:

Problem 1. *Given a directed graph $G = (V, E)$, where each node v is associated with attributes $F_v = \{f_v^1, f_v^2, \ldots\}$ from a universe of eligible attributes Φ, a seed set of adopter nodes S, quantities q_{uv}, b_{uv} for each edge $e_{uv} \in E$, and a transition function $h_{uv}(F_v, F) = \min\left\{\frac{1-b_{uv}}{q_{uv}}, |F_v \cap F|\right\}$ for edge probabilities, select a set of k attributes $F \subset \Phi$ that maximizes the spread $\sigma(F|S)$ of a diffusion process with attributes F starting from S.*

APIM is a novel problem that aims to find out how one can maximize the benefits of a network promotion campaign with given points of departure. The motivation derives from the fact that, in the real world, brands want to exploit their own social network pages for marketing purposes. Instead of targeting the most influential initiators for a promotion, as in classical IM, one can judiciously invest in the creation of a post with lucrative content, under fixed initiators, guided by the content attributes provided by the APIM solution. As promotions can be formed with a wide variety of selected attributes, each possible attribute set F corresponds to a different probabilistic graph, on which we can compute the influence spread of the seed set S; the attribute set F that achieves maximum spread constitutes the APIM solution. We emphasize that, due to the drastic difference between classical IM and APIM in terms of the way influence spread is achieved, *the solutions to these two problems cannot be qualitatively compared against each other.*

4 HARDNESS AND INAPPROXIMABILITY

We now show the hardness of the APIM problem and study the properties of influence spread function $\sigma(F)$. To calculate $\sigma(F)$, we first calculate edge probabilities with respect to the selected attributes F and then estimate the expected spread on the graph starting from the given set of subscribed nodes S.

Theorem 4.1. *The APIM problem is NP-hard for the content-aware cascade model.*

Proof. Consider an instance of the NP-complete *SET COVER* problem, defined by a collection of subsets $S_1, S_2, \ldots, S_m$, a universe of elements $U = \{u_1, u_2, \ldots, u_n\}$ and an integer k. We are asked whether there are k sets that will cover all elements in U. We show that SET COVER can be reduced to a *trivial* instance of APIM as follows: We construct a bipartite graph with one activated node on the left side that connects to n nodes on the right side, as shown in Figure 1. We map each member u_i of universe U to a node on the right side and add an attribute f_j to set F_{u_i} if u_i belongs to subset S_j. We set $b_{uv} = 0$ and $q_{uv} = 1$ for all edges $(u, v) \in E$, i.e., a node v is influenced if at least one of its attributes is selected. We emphasize that, in this trivialized version of APIM, there spread can be computed deterministically; there is no need for expected spread computations. Then, an algorithm that could optimally solve this trivial instance of APIM, among others, would decide any instance of SET COVER: if we can target all nodes in the APIM instance using k attributes, we can in effect cover all elements in U using k subsets in SET COVER. Otherwise, if the optimal spread in APIM does not reach all nodes, it follows that there is no set of k subsets that covers all elements in SET COVER. Thus, by reduction from SET COVER, APIM is at least as hard as any problem in NP. □

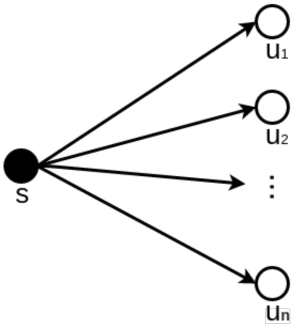


Figure 1: A graph instance demonstrating that the APIM problem is NP-hard.

By Theorem 4.1, there is no polynomial-time algorithm to find an optimal set of attributes F, unless P=NP. We now proceed to study function $\sigma(F)$.

We call a transition function $h_{uv}(F_v, F)$ monotonic on F if for subsets of attributes $F_1 \subset F_2 \subset \Phi$, it holds that $h_{uv}(F_v, F_1) \leq h_{uv}(F_v, F_2)$ for any node v. If the transition function is not monotone, then the influence spread function is neither monotone, nor submodular, because selecting more attributes may reduce probabilities $p(u, v)$ and thereby reduce the total influence spread. In the following, we study the properties of influence function $\sigma(F)$ under the assumption that attributes can only have a positive or neutral effect on users, thereby rendering the transition function $h_{uv}(\cdot)$ monotonic. For such a monotonic transition function, edge probabilities can only increase if we add attributes to F, i.e. $h_{uv}(F_v, F) \leq h_{uv}(F_v, F + \{f\})$ for any $f \in \Phi$ and $v \in G$. Therefore, our influence spread function $\sigma(F)$ is monotonic. We now proceed to examine whether the objective influence spread function in the APIM problem is submodular.

A function $\sigma(F)$ is *submodular* if it follows a *diminishing returns* rule: the marginal gain from adding an element to a set F is at most as high as the marginal gain from adding the same element to a subset of F. That is, $\sigma(F_1 \cup \{f\}) - \sigma(F_1) \geq \sigma(F_2 \cup \{f\}) - \sigma(F_2)$, where $F_1 \subset F_2$ and any attribute $f \in F$.

We examine whether the influence function $\sigma(F)$ in the APIM problem is submodular. Unfortunately, it turns out that this is not the case, even for a monotone and submodular transition function, as the following counterexample demonstrates.

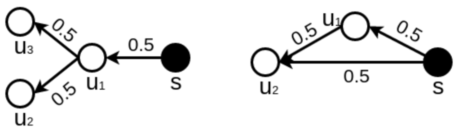


Figure 2: Increasing & decreasing marginal returns

Example 4.2. Consider the graph on the left-hand side in Figure 2, with a universe of attributes $\Phi = \{A, B, C\}$, sets of preferred attributes for each node be $F_{v_1} = \{A\}$ and $F_{v_2} = F_{v_3} = \{A, B, C\}$, $b_{uv} = 0.5$, $q_{uv} = \frac{1}{2|F_v|}$ on all edges, and one active node s. Then, consider two subsets of attributes $F_1 = \emptyset$, $F_2 = \{B, C\}$, where $F_1 \subset F_2$, and a attribute $f = A \in \Phi \setminus F_2$. The achieved spreads for each attributes subset, and the respective marginal gains obtained after adding attribute f to subsets F_1 and F_2, are calculated as follows. For subset attribute F_1 selected, we have:

$$p_{sv_1} = \frac{1}{2}, \quad p_{v_1v_2} = p_{v_1v_3} = \frac{1}{2}$$
$$\sigma(F_1) = \frac{1}{2} + 2\frac{1}{4} = 1$$

whereas when $f = A$ is added to F_1 we get:

$$p_{sv_1} = 1, \quad p_{v_1v_2} = p_{v_1v_3} = \frac{2}{3}$$
$$\sigma(F_1 + \{A\}) = 1 + 2 \cdot \frac{2}{3} = \frac{7}{3}$$

Hence $\Delta_1 = \sigma(F_1 + \{A\}) - \sigma(F_1) = \frac{7}{3} - 1 = \frac{4}{3}$. Similarly, for F_2 selected, we have:

$$p_{sv_1} = \frac{1}{2}, \quad p_{v_1v_2} = p_{v_1v_3} = \frac{5}{6}$$
$$\sigma(F_2) = \frac{1}{2} + 2 \cdot \frac{5}{12} = \frac{4}{3}$$

while when $f = A$ is added to F_2 we get:

$$p_{sv_1} = 1, \quad p_{v_1v_2} = p_{v_1v_3} = 1$$
$$\sigma(F_2 + \{A\}) = 3$$

Hence $\Delta_2 = \sigma(F_2 + \{A\}) - \sigma(F_2) = 3 - \frac{4}{3} = \frac{5}{3}$. Since $\Delta_2 > \Delta_1$, the submodularity of $\sigma(F)$ does not hold.

Given this negative results, the influence function $\sigma(F)$ might have an *increasing returns* property (supermodularity), whereby it would hold that $\sigma(F_1 \cup \{f\}) - \sigma(F_1) \leq \sigma(F_2 \cup \{f\}) - \sigma(F_2)$, for $F_1 \subset F_2$ and any attribute $f \in F$. The following counterexample shows that this property does not hold either.

Example 4.3. Consider the graph on the right-hand side in Figure 2, with a universe of attributes $\Phi = \{A, B\}$, sets of preferred attributes per node $F_{v_1} = \{A, B\}$ and $F_{v_2} = \{A\}$, $b_{uv} = 0.5$ and $q_{uv} = \frac{1}{2|F_v|}$ on all edges, and one active node s. Then, consider two subsets of attributes $F_1 = \emptyset$ and $F_2 = \{B\}$. Then, for subset attribute F_1 selected, we have:

$$p_{sv_1} = \frac{1}{2}, \quad p_{sv_2} = p_{v_1v_2} = \frac{1}{2}$$
$$\sigma(F_1) = \frac{1}{2} + (1 - (1 - \frac{1}{4})\frac{1}{2}) = \frac{9}{8}$$

whereas when $f = A$ is added to F_1 we get:

$$p_{sv_1} = \frac{3}{4}, \quad p_{sv_2} = p_{v_1v_2} = 1$$
$$\sigma(F_1 + \{A\}) = \frac{3}{4} + 1 = \frac{7}{4}$$

Hence $\Delta_1 = \sigma(F_1 + \{A\}) - \sigma(F_1) = \frac{7}{4} - \frac{9}{8} = \frac{5}{8}$. Similarly, for F_2 selected, we have:

$$p_{sv_1} = \frac{3}{4}, \quad p_{sv_2} = p_{v_1v_2} = \frac{1}{2}$$
$$\sigma(F_2) = \frac{3}{4} + (1 - (1 - \frac{3}{8})\frac{1}{2}) = \frac{23}{16}$$

while when $f = A$ is added to F_2 we get:

$$p_{s,v_1} = 1, \quad p_{v_1v_2} = p_{v_1v_2} = 1$$
$$\sigma(F_2 + \{A\}) = 2$$

Hence $\Delta_2 = \sigma(F_2 + \{A\}) - \sigma(F_2) = 2 - \frac{23}{16} = \frac{9}{16}$. Since $\Delta_1 > \Delta_2$, the influence function $\sigma(F)$ is not supermodular either.

Eventually, we have established the following:

Theorem 4.4. *The spread function $\sigma(F)$ with a probability transition function $h_{uv}(F_v, F) = \min\left\{\frac{1-b_{uv}}{q_{uv}}, |F_v \cap F|\right\}$ is neither submodular nor supermodular.*

By Theorem 4.4 it follows that we cannot use a greedy algorithm with approximation guarantee based on submodularity, as in [27]. Moreover, in the following we show that it is NP-hard to approximate the optimal solution to APIM.

Theorem 4.5. *It is NP-hard to approximate the optimal solution to the APIM problem for Independent Cascade model within a factor $n^{1-\varepsilon}$ for any $\varepsilon > 0$.*

Proof. Consider an instance of the SET COVER problem, in which we need to decide whether we can cover all elements of a universe $U = \{u_1, u_2, \ldots, u_n\}$ by selecting at most k subsets out of a collection of $S_1, S_2, \ldots, S_m \subset U$.

We then construct a graph G for the APIM problem with a single subscriber node s and nodes $u_1, u_2, \ldots, u_n$ corresponding to elements in U, connected so that u_{i-1} points towards u_i for all $i = 2 \ldots n$, and s is connected to u_1, and, for every subset S_i an element u_j belongs to, we add a attribute f_i to the preferred attributes of u_j. Next, for some integer c we add $\eta = n^c - n - 1$ more nodes $x_1, x_2, \ldots, x_\eta$ such that u_n has outgoing edges to them and each x_i has the same preferred attributes as u_n. Graph G, shown in Figure 3, has $N = n^c$ nodes. We set $b_{uv} = 0$ and $q_{uv} = 1$ for all edges, so that an edge becomes active if at least one of the attributes associated with its target node is selected. Then, if it is possible to select k subsets that cover all elements of universe U, we can also have $N = n^c$ activated nodes. Conversely, if there is no selection of k subsets that covers all U, then there is at least one node u_i that does not get activated, precluding influence spread to nodes $x_1, x_2, \ldots, x_\eta$. We can then only target at most n out of n^c nodes, a fraction of $n^{1-c} = N^{\frac{1}{c}-1}$. Consequently, if we had a polynomial algorithm that approximates the optimal solution to APIM within a factor of $N^{1-\varepsilon}$ for any ε, then it would suffice to set $c = \left\lceil \frac{1}{\varepsilon} \right\rceil$ and use that algorithm so as to decisively distinguish between a case that accepts a solution activating all N nodes and one that does not, and thereby also decide SET COVER. Thus, by reduction, we have shown that it is NP-hard to approximate the optimal solution to APIM within a reasonable factor. □

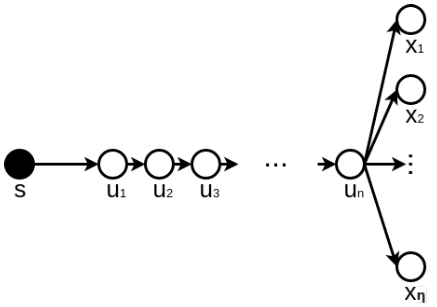


Figure 3: A graph instance demonstrating that it is NP-hard to approximate the optimal solution to the APIM problem.

5 THE EXPLORE-UPDATE ALGORITHM

As the APIM problem cannot be approximated within a factor $n^{1-\varepsilon}$ with the content-aware cascade model, we proceed to design heuristic solutions therefor. We structure our exposition as follows: we first present a simple, yet time-consuming greedy heuristic; then, through a sequence of simplifying assumptions, we will generate a much more efficient algorithm called Explore-Update.

Algorithm 1: Greedy(G, S, k)

```
F = ∅;
while |F| < k do
    for every f ∈ Φ \ F do
        calculate σ(F + {f}) using Monte Carlo simulations
    F = F ∪ argmax_f {σ (F + {f})}
return F
```

Our first proposal is a baseline greedy algorithm that selects the attribute of highest marginal gain to add at each iteration, shown in Algorithm 1. This is an adaptation of the *Local Update* algorithm in [6] to our problem. Intuitively, it is reasonable to greedily select the locally best attribute in each iteration, especially for small values of k. This kind of algorithm has been shown to achieve better quality than others in classical Influence Maximization [11, 13, 28].

Though simple and effective, Algorithm 1 is inefficient due to its calculation of influence spread by MC simulations. In a manner reminiscent of [13], we can improve efficiency by considering *maximum influence paths* between nodes and the seed set. We call a path $P_{max} = \langle u = u_0, u_1, u_2, \ldots, v = u_m \rangle$ between vertices $u \in S$ and $v \in G$ *maximum influence path* (MIP) if this path is the most probable among all paths between u and v: $P_{max} = \operatorname{argmax}_P \prod_{i=0}^{m-1} prob(u_i, u_{i+1})$. Under the *simplifying assumption* that influence is propagated only through MIPs, we can estimate influence spread in polynomial time as follows: For a threshold θ and a node v, we build a tree structure called *in-arborescence* $A_{in}(v)$, which includes all MIPs of probability higher than θ from any node to v: $A_{in}(v) = \{\mathsf{MIP}(u, v) \mid prob(\mathsf{MIP}(u, v)) > \theta, u \in G\}$. Then, given a node u, the seed set S, and an arborescence $A_{in}(v)$, Algorithm 2 recursively estimates the probability that u is activated in $A_{in}(v)$, i.e., its *activation probability* $ap(u, A_{in}(v))$.

Algorithm 2: calculateAP(u, $A_{in}(v)$, S)

```
if $u \in S$ then
    $ap(u, A_{in}(v)) = 1$
else if $u$ has no in-neighbors in $A_{in}(v)$ then
    $ap(u, A_{in}(v)) = 0$
else
    $ap(u, A_{in}(v)) = 1 - \prod_{\omega \in N_{in}(u)} (1 - ap(\omega, A_{in}(v)) prob(\omega, u))$
return $ap(u, A_{in}(v))$
```

Based on these calculations, for all nodes $u \in G$, we can calculate the influence spread $\sigma(F)$ as follows:

$$\sigma(F) = \sum_{u \in G} ap(u, A_{in}(u)) \tag{3}$$

We can then employ Equation (3) so as to estimate influence spread in Algorithm 1, in lieu of MC simulations, deriving Algorithm 3; at each iteration, we compute the in-arborescence of node u for a given threshold θ by converting each probability p_e on an edge e to $-\log p_e$ and employing an efficient implementation of Dijkstra's algorithm. If computing an arborescence takes time t, then Lines 4-6 take nt and the total time is $O(k|\Phi|nt)$.

Algorithm 3: Arb(G, S, θ, k)

```
1 $F = \emptyset$;
2 while $|F| < k$ do
3     for every $f \in \Phi \setminus F$ do
4         for every $u \in G$ do
5             compute $A_{in}(u)$ with threshold $\theta$
6             $ap(u, A_{in}(u))$ = calculateAP($u$, $A_{in}(u)$, $S$)
7         calculate $\sigma(F + f) = \sum_{u \in G} ap(u, A_{in}(u))$
8     $F = F \cup \operatorname{argmax}_f \{\sigma(F + \{f\})\}$
9 return $F$
```

We further reduce the runtime of Algorithm 3 by eschewing redundant iterations of the loops over nodes u and attributes f. First, we limit the calculation of in-arborescences and activation probabilities only to nodes whose in-arborescence under threshold θ reaches at least one node in S; only such nodes can yield non-zero estimated activation probability. To find out these nodes, we compute the out-arborescence of all nodes in S, $A_{out}(S)$, consisting of all MIPs of probability higher than θ from a node $v \in S$ to other nodes in G. Nodes in $A_{out}(S)$ yield non-zero activation probability estimates. Yet the set of paths in $A_{out}(S)$ may contain directed loops, hence we cannot apply a recursive algorithm like Algorithm 2 directly on $A_{out}(S)$; we still need to obtain the in-arborescence $A_{in}(u)$ of each $u \in A_{out}(S)$; we do so while building $A_{out}(S)$, by adding MIP(v, u) to $A_{in}(u)$ for each $u \in A_{out}(v)$. Algorithm 4 illustrates this Explore process.

Algorithm 4: Explore(G, F, S, θ)

```
$A_{out}(S) = \emptyset$
$A_{in}(u) = \emptyset$ for every $u$ in $G$
for every $v \in S$ do
    compute $A_{out}(v)$ for given $\theta$ and $F$
    update $A_{in}(u)$ for each $u \in A_{out}(v)$
return $\{A_{in}(u) \neq \emptyset | \text{ for } u \in G\}$
```

Then we can calculate influence spread $\sigma(F)$ using the union of such in-arborescences, A_{in}, by Algorithm 5.

Algorithm 5: Update(A_{in}, S)

```
for every $u \in A_{in}$ do
    $ap(u)$ = CalculateAP($u$, $A_{in}(u)$, $S$)
$\sigma(F) = \sum_{u \in A_{in}} ap(u)$
return $\sigma(F)$
```

Second, we limit the calculation of marginal gain in Algorithm 3 only to those attributes that can affect the influence spread. We call an edge in G *participating*, if at least one of its endpoints are in $A_{out}(S)$. Figure 4 presents a graph for a seed set S (and selected attributes set F) in the green area; the yellow area includes nodes in $A_{out}(S)$; the set of participating edges Π is shown in solid and dotted lines; dotted edges have only one endpoint in $A_{out}(S)$; non-participating edges are shown in dashed lines, in the gray area.

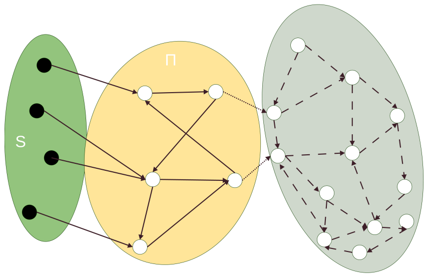


Figure 4: Participating and non-participating edges

Non-participating edges cannot increase influence spread, regardless whether their probability is increased; only participating edges have such potential. We limit the attributes Algorithm 3 considers based on this observation. Let $E(f)$ be the set of edges that include attribute f among their preferred attributes, hence their probability is affected when adding f to F. Then, at any iteration, if *none* of the edges in $E(f)$ is a participating edge, i.e., $E(f) \cap \Pi = \emptyset$, then attribute f need not be examined as a candidate to be added to F; it bears no effect to influence function $\sigma(F + \{f\})$.

Putting together our enhancements to Algorithm 3, we design the polynomial-time Explore-Update algorithm (Algorithm 6). In a nutshell, at each iteration, this Explore-Update algorithm selects the hitherto unselected attribute f affecting participating edges that brings about the largest increase of influence spread, using the Explore procedure for calculating in-arborescences and the Update procedure for calculating influence spread, while updating the set of participating edges Π at each iteration and using it to determine which attributes need to be examined at the next iteration.

Algorithm 6: Explore-Update(G, S, k, θ)

```
$F = \emptyset$
$A_{in}$ = Explore($G$, $F$, $S$, $\theta$)
$\Pi = \{(u, v) \in G | u \in A_{in} \text{ or } v \in A_{in}\}$
while $|F| < k$ do
    for $f \in \Phi \setminus F$ do
        if $E(f) \cap \Pi \neq \emptyset$ then
            $A_{in}$ = Explore($G$, $F + \{f\}$, $S$, $\theta$)
            $\Pi_f = \{(u, v) \in G | u \in A_{in} \text{ or } v \in A_{in}\}$
            $\sigma(F + \{f\})$ = Update($A_{in}$, $S$)
    $f_{max} = \text{argmax}_f \{\sigma(F + \{f\})\}$
    $F = F \cup f_{max}$
    $\Pi = \Pi_{f_{max}}$
return $F$
```

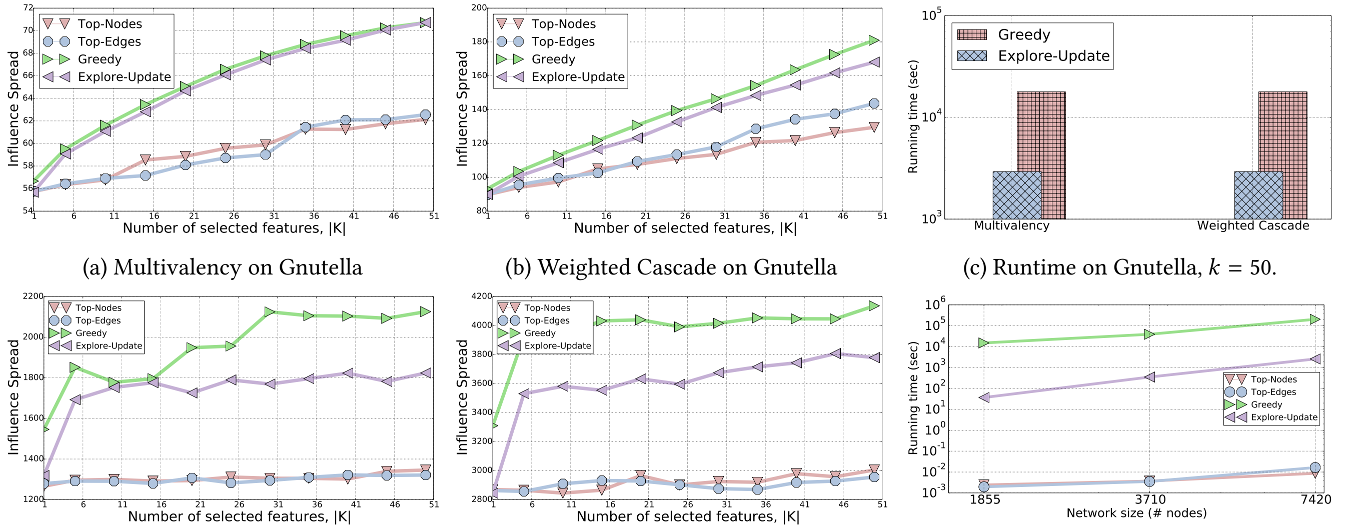
Let the time complexity to calculate an out-arborescence for node in S be $t_{out\theta}$, then the Explore procedure takes $|S|t_{out\theta}$ and the Update procedure takes $O(n_{in\theta}n_{out\theta})$ time, where $n_{in\theta}$ is the number of nodes in in-arborescences, and $n_{out\theta}$ is the number of nodes in out-arborescence of S. Therefore, if we perform κ calculations of A_{in} per iteration, the total runtime is $O(k\kappa(|S|t_{out\theta} + n_{in\theta}n_{out\theta}))$. In effect, the Explore-Update algorithm is expected to perform well when the size of arborescences is small, and the number of updates κ per iteration is smaller than $|\Phi|$. As propagation probabilities on edges are usually small in real networks, the size of arborescences is indeed expected to be small. The number of updates depends on the structure of the network. In a large-diameter network where multiple hops are required to reach most nodes from S via a MIP, there is a good chance to reduce the number of computations significantly. We investigate this matter experimentally in the following.

6 EXPERIMENTAL STUDY

In this section we present a comprehensive experimental study on the Greedy and Explore-Update algorithms we have introduced. All experiments were run on a 32GB Intel Core i5-2450M CPU machine @ 2.50GHz, while algorithms were implemented in C++.

As there is no previous work on the APIM problem, we compare to basic baselines. Still, as we discussed, the previous work that comes closest to our problem is that by Barbieri and Bonchi [6]; still, that work solves primarily the problem of selecting a set of seed nodes, and secondarily a set of product attributes, so as to maximize product influence in a network. The best-performing algorithm for updating an attribute set in [6], *Local Update*, performs one addition or removal of an attribute to/from the current attribute set at each iteration; in effect, our Greedy algorithm can be considered as an adaptation of *Local Update* to our problem, where only additions of attributes are needed. We emphasize that, to the extent that a comparison to [6] is possible, *we conduct it via the comparison to the Greedy algorithm itself.* Another method for updating an attribute set proposed in [6], *Generic Update*, is a hard-to-tune genetic algorithm, which may lead to an unpredictable number of output attributes. Besides, as the experimental study in [6] shows, *Genetic Update* offers no qualitative advantage while it is much slower than *Local Update*, which is already by far the most time-consuming algorithm in our study. Therefore, we do not consider a genetic algorithm in our experimental study.

Diffusion models. In the content-aware cascade model the probability on edge (u, v) is a linear function of product and base



(a) Multivalency on Gnutella

(b) Weighted Cascade on Gnutella

(c) Runtime on Gnutella, $k = 50$.

(d) Multivalency on VK

(e) Weighted Cascade on VK

(f) Runtime on VK with Multivalency, $k = 20$.

Figure 5: Influence spread and runtime results.

probabilities q_{uv} and b_{uv}. To assign these probabilities we use two techniques prevalent in previous work [13].

- Weighted Cascade model: probability $\frac{1}{d_v}$ is assigned to edge (u, v), where d_v is the in-degree of node v. We use this model for the sake of compatibility with previous works, even while it may fit less to our problem setting.
- Multivalency model: the probability for edge (u, v) is drawn uniformly at random from a set of probabilities. We choose that set to be $[0.02, 0.04, 0.08]$.

We calculate b_{uv} for every edge (u, v), and set $q_{uv} = \frac{b_{uv}}{|F_v|}$.

Algorithms. We compare the Explore-Update algorithm under different threshold θ values to three other algorithms:

- **Greedy** This is Algorithm 1 in this paper, which is effectively an adaptation of *Local Update*, the best algorithm in [6]. A similar algorithm has been used extensively in the context of the Influence Maximization problem, and always demonstrated top performance in terms of spread, while being slower than other heuristics [11]; it requires specifying the number of Monte-Carlo simulations to calculate influence spread, as we do in the following.
- **Top-Nodes** This algorithm measures each attribute's frequency among node preferences and selects the k most frequent ones.
- **Top-Edges** This algorithm assigns to each edge $e = (u, v)$ the attribute preferences of node v, F_v, and select the k most frequent attributes across all edges.
- **Brute-Force** This algorithm finds all possible sets of attributes of size k, computes each one's influence spread using Monte-Carlo simulations, and opts for the best. Because the solution space is exponential, we use this method on reduced datasets.

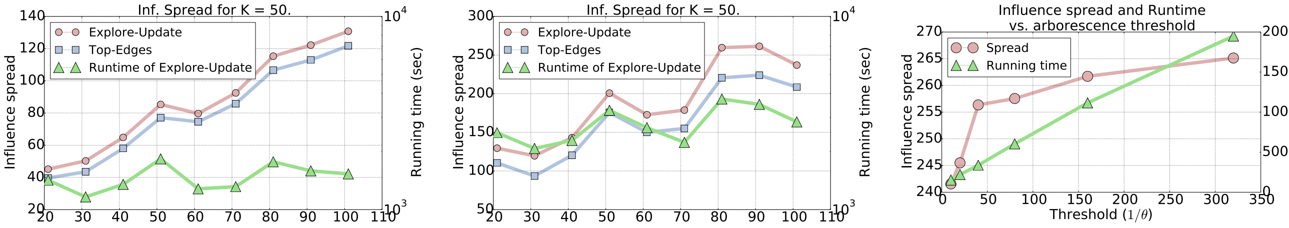
Datasets. We run experiments in two real-world networks. The first network is a peer-to-peer file sharing directed network Gnutella[1], where nodes represent hosts and edges represent connections between the Gnutella hosts. Our second network is extracted by crawling the social network VK[2]; nodes are users and edges are friendships among them. Statistics are presented in Table 1.

Dataset	Gnutella	VK
Nodes	10,876	7,420
Edges	39,994	57,638
Average Clustering Coefficient	0.0062	0.28
Number of Triangles	934	168,284
Diameter	9	16
Attributes/Seed sets	151	3,882
Default Seed Size	34	15

Table 1: Data characteristics

Attribute assignment and seed selection. We utilize one general and one ad-hoc method for attribute preference assignment. In Gnutella, to assign an attribute preferences set F_v to node v, we find the block partitioning that minimizes the description length of the network by stochastic blockmodel ensemble; this technique is used to discover the block structure of empirical networks and results to block memberships for each node [25, 32]. We allow nodes to have overlapping memberships to different blocks. Each block β_i is associated with a distinct attribute f_i. The attribute preference set of a node v_j, F_{v_j} is the set of attributes of the blocks v_j belongs to. The returned partitioning consists of 151 blocks; the default seed set S is one of the blocks, of size 34. For VK, the data comes along with annotations of *groups* and *pages*, which allow us to derive both node attributes and seed sets. A *group* or *page* is a community of users that share content with each other and communicate about a topic of interest (e.g., football clubs or TV series). We use these group memberships to derive both node attributes and seed sets, consistently to our motivation. There are 3882 such groups; the default seed size is 15. Unless otherwise indicated, in our experiments we use the default seeds.

[1] https://snap.stanford.edu/data/p2p-Gnutella04.html
[2] https://vk.com/



(a) vs. seed set size, MV

(b) vs. seed set size, WC

(c) vs. θ

Figure 6: Influence and runtime vs. seed set size and θ on Gnutella.

6.1 Influence spread

Figures 5(a-b) present our results on competing algorithms' influence spread[3] on the Gnutella network, varying number of selected attributes k from 1 to 51. We used 10000 MC simulation for Greedy, and $\theta = 1/320$ for Explore-Update. We observe that Explore-Update arrives just 1% and 5% below the performance of Greedy with the Multivalency and Weighted Cascade model, respectively. On the other hand, the Top-Edges and Top-Nodes algorithms reach only 88% and 85% of the spread of Explore-Update. Figures 5(d-e) present influence spread in VK network. We use Greedy with 500 Monte-Carlo simulations, achieving 15%, 37%, and 48% higher spread than Explore-Update with $\theta = 1/40$, Top-Nodes, and Top-Edges, respectively, in MV model. The picture is similar with the WC model, where Greedy achieves spread 9%, 38%, 40% higher than Explore-Update, Top-Nodes, and Top-Edges. Overall, our results confirm that Explore-Update achieves high influence spread for networks where the local neighborhood of the seed set has structure amenable to long distance arborescences.

6.2 Runtime

We now compare algorithms in terms of runtime. Figure 5(c) presents the results with Gnutella for $k = 50$; Explore-Update ($\theta = 1/320$) runs an order of magnitude faster than Greedy (10000 iterations); Top-Edges and Top-Nodes output a selected set in less than a second, hence we do not include them. Next, we investigate how the algorithms scale with increasing network size. We extract subnetworks of VK consisting of 1855, 3710, and 7420 nodes of the original network (i.e., 1/4, 1/2, and full network) and proportional edge density to the full network. In all cases, we compute the runtime on a seed set S of size 15, with the Multivalency model for $k = 20$, for Greedy (10000 iterations), Explore-Update ($\theta = 1/40$), and the Top-Edges and Top-Nodes heuristics. Figure 5(f) shows that runtime scales linearly in network size in all cases. Moreover, we ascertain that while Explore-Update fares no better than Greedy in terms of influence spread, it is much faster.

6.3 Effect of Seed Size

We now test the performance of Explore-Update for different sizes of the seed set S. We select different seed sets from size 21 (minimal size for the current block partition) to 101 with step 10 on Gnutella. Figure 6(a-b) presents the influence spread for Explore-Update and Top-Edges for $k = 50$, as well as the runtime of Explore-Update, whereas Greedy is orders of magnitude slower for this setup, and Top-Nodes performs worse than Top-Edges. We note that Explore-Update always achieves better influence spread than Top-Edges. Interestingly, influence spread and runtime do not always grow with $|S|$. This is explicable by the fact that different seed sets induce different local structures.

6.4 Effect of θ

Next, we study the effect of the θ threshold, which controls the size of arborescences and thereby the influence spread achievable from seed set S. Figure 6(c) presents the influence spread and runtime with the Gnutella network for θ in $\{\frac{1}{10}, \frac{1}{20}, \dots, \frac{1}{320}\}$, with the WC model for $k = 50$. The runtime of Explore-Update grows linearly in the inverse threshold θ, while influence spread grows logarithmically in it. A good tradeoff between quality and runtime is found at the knee point in the influence spread curve for $\theta = \frac{1}{40}$.

6.5 Comparison to the Optimal Solution

By Theorem 4.5, we proved it is NP-hard to approximate the optimal solution to APIM. Now, we compare the results of heuristics to the optimal solution obtained by brute force; we reduce the total number of attributes to 16 and use a reduced Gnutella network by selecting 2K nodes, yielding similar degree distribution properties to the original. Figure 7 shows the influence spread results, with the Multivalency model, for a random seed set of size 10. Remarkably, Explore-Update finds the optimal set of attributes with varying k.

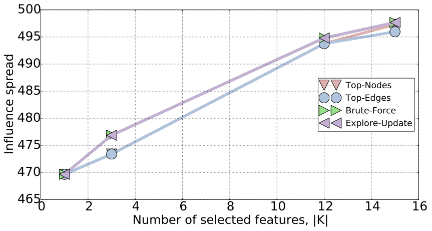


Figure 7: Influence spread on reduced network.

Next, we select $k = 10$, yielding $\binom{16}{10} = 8008$ possible attribute sets, and calculate, with a new random seed set of size 428, the *rank* of each algorithm's solution among all possible attribute sets: for each attribute set, we compute its influence spread using 10000 MC simulations; we sort sets by their spread values, and identify the rank of the solution returned by each heuristic. Table 2 presents those ranks. Explore-Update selects the optimal solution, while Greedy with 500 MC iterations yields the fifth best attribute set. The selected attribute sets differ from each other in 2 out of 10 attributes. We obtained similar results for other values of k, with Explore-Update always returning the optimal attribute set.

[3] We use 10000 Monte-Carlo simulations to compute the final spread of all solutions.

Algorithm	Rank	Spread
Explore-Update	1	34.592
Greedy	5	34.114
Top-Edges	113	33.592
Top-Nodes	113	33.592
. . .		
–	8008	27.82

Table 2: Algorithm ranking w.r.t. optimal solution.

6.6 Real-World Examples

Last, we looked into the actual results - seed sets and selected attribute sets of our experiments, with special attention to the VK data set with the multivalency model, and inspected our results. One interesting observation was that those attributes that are liked by seed set users were rarely among the ones selected in the final solution; this fact indicates that our problem makes good practical sense, while a straightforward naive solution of sticking to what is liked by seed nodes does not yield good results. Nevertheless, selected attributes exhibited a remote, yet unpredictable, resemblance to the attributes liked by seed set nodes. For example, with a group titled “La vie et l’amour” as seed, the selected attributes in our VK network sample included “Home Comfort — Design — Interior Design — Style”. With “Psychology of Relations” as seed, the selected attribute set included “Philosophy of Life”. Such analogies between seed set and selected attributes, while retrospectively intuitive, would not be derived otherwise; they depend on the way nodes of diverse interests interact within the overall network structure. Such results vindicate our problem motivation.

We also checked how result sets change when we vary k. For example, we select 100 out of 431,374 subscribers of “Esoterica YOGA MEDITATION” as seed set. With $k = 3$, the selected attributes are “MODA”, “La vie et l’amour”, “Blog for Men”. As “Esoterica YOGA MEDITATION” targets primarily women, results such as “MODA” and “La vie et l’amour” are unsurprising. Nevertheless, interestingly, both E-U and Greedy also return “Blog for Men” as a selected attribute, whereas the simple Top-Nodes and Top-Edges heuristics do not. This result shows that our algorithm can select nontrivial attributes.

7 CONCLUSION

This paper proposed the problem of content recommendation for viral social influence. The goal is to select k attributes that characterize a propagated meme’s content, such that its spread across a network from given points of departure is maximized, whereby different attribute sets yield different network edge probabilities. To our knowledge, there is no previous work on this problem. We formulated a content-aware cascade model and showed that our problem is NP-hard and inapproximable, while the influence function is neither submodular, nor supermodular. We developed an efficient algorithm for the problem, using bounded local arborescences to calculate influence spread. Our experimental study demonstrates that this Explore-Update algorithm selects topics sets that achieve high spread and is orders of magnitude faster than a conventional Greedy solution resembling algorithms developed for related problems. We also provide evidence that our E-U algorithm can achieve the optimal solution when the number of selected topics is small. In the future, we plan to study other propagation models and investigate the parallelization of Explore-Update.

REFERENCES

[1] S. Aral. 2011. Commentaty-Identifying social influence: A comment on opinion leadership and social contagion in new product diffusion. *Marketing Science* 30, 2 (2011), 217–223.

[2] S. Aral, L. Muchnik, and A. Sundararajan. 2009. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proc. of the National Academy of Sciences of the U.S.A.* 106, 51 (2009), 21544–21549.

[3] S. Aral and D. Walker. 2011. Creating social contagion through viral product design: A randomized trial of peer influence in networks. *Management Science* 57, 9 (2011), 1623–1639.

[4] C. Aslay, N. Barbieri, F. Bonchi, and R. A. Baeza-Yates. 2014. Online topic-aware influence maximization queries. In *EDBT*.

[5] C. Aslay, W. Lu, F. Bonchi, A. Goyal, and Laks V.S. Lakshmanan. 2015. Viral marketing meets social advertising: Ad allocation with minimum regret. *Proc. VLDB Endow.* 8, 7 (2015), 814–825.

[6] N. Barbieri and F. Bonchi. 2014. Influence maximization with viral product design. In *SDM*.

[7] N. Barbieri, F. Bonchi, and G. Manco. 2012. Topic-aware social influence propagation models. In *ICDM*.

[8] J. A. Berger and C. Heath. 2005. Idea Habitats: How the prevalence of environmental cues influences the success of ideas. *Cognitive Science* 29, 2 (2005), 195–221.

[9] C. Borgs, M. Brautbar, J. T. Chayes, and B. Lucier. 2014. Maximizing social influence in nearly optimal time. In *SODA*.

[10] S. Chen, J. Fan, G. Li, J. Feng, K.-L. Tan, and J. Tang. 2015. Online topic-aware influence maximization. *Proc. VLDB Endow.* 8, 6 (2015), 666–677.

[11] Wei Chen, Laks V. S. Lakshmanan, and Carlos Castillo. 2013. *Information and Influence Propagation in Social Networks.* Morgan & Claypool Publishers.

[12] W. Chen, T. Lin, and C. Yang. 2014. Efficient topic-aware influence maximization using preprocessing. *CoRR* abs/1403.0057 (2014).

[13] W. Chen, C. Wang, and Y. Wang. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*.

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

[15] Y. -C. Chen, W. -C. Peng, and S. -Y. Lee. 2012. Efficient algorithms for influence maximization in social networks. *Knowl. Inf. Syst.* 33, 3 (2012), 577–601.

[16] S. Cheng, H. Shen, J. Huang, W. Chen, and X. Cheng. 2014. IMRank: influence maximization via finding self-consistent ranking. In *SIGIR*.

[17] I. P. Cvijikj and F. Michahelles. 2013. Online engagement factors on Facebook brand pages. *Social Network Analysis and Mining* 3, 4 (2013), 843–861.

[18] Mahashweta Das, Habibur Rahman, Gautam Das, and Vagelis Hristidis. 2013. Generating informative snippet to maximize item visibility. In *CIKM*. 1721–1726.

[19] L. de Vries, S. Gensler, and Peter S.H. Leeflang. 2012. Popularity of brand posts on brand fan pages: An Investigation of the Effects of Social Media Marketing. *Journal of Interactive Marketing* 26, 2 (2012), 83–91.

[20] C. Van den Bulte and G. L. Lilien. 2001. Medical innovation revisited: Social contagion versus marketing effort. *Am. J. Sociol.* 106, 5 (2001), 1409–1435.

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

[22] D. Godes and D. Mayzlin. 2004. Using online conversations to study word-of-mouth communication. *Marketing Science* 23, 4 (2004), 545–560.

[23] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. 2011. A data-based approach to social influence maximization. *PVLDB* 5, 1 (2011), 73–84.

[24] L. Hong and B. D. Davison. 2010. Empirical study of topic modeling in twitter. In *SOMA*.

[25] Brian Karrer and M. E. J. Newman. 2011. Stochastic blockmodels and community structure in networks. *Phys. Rev. E* 83 (Jan 2011), 016107. Issue 1.

[26] E. Katz. 1959. Mass communications research and the study of popular culture: An editorial note on a possible future of this journal. *Studies in Public Communication* 2 (1959), 1–6.

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

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

[29] Y. Li, D. Zhang, and K.-L. Tan. 2015. Real-time targeted influence maximization for online advertisements. *Proc. VLDB Endow.* 8, 10 (2015), 1070–1081.

[30] C. F. Manski. 1993. Identification of endogenous social effects: The reflection problem. *The Review of Economic Studies* 60, 3 (1993), 531–542.

[31] Muhammed Miah, Gautam Das, Vagelis Hristidis, and Heikki Mannila. 2009. Determining Attributes to Maximize Visibility of Objects. *IEEE Trans. Knowl. Data Eng.* 21, 7 (2009), 959–973.

[32] Tiago P. Peixoto. 2015. Model Selection and Hypothesis Testing for Large-Scale Network Models with Overlapping Groups. *Phys. Rev. X* 5 (2015), 20. Issue 1.

[33] M. Richardson and P. Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *KDD*.

[34] Y. Tang, X. Xiao, and Y. Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*.