Viral vs. Effective: Utility Based Influence Maximization

Yael Sabato
Computer Science Department
Ariel University, Israel
yael.sabato@msmail.ariel.ac.il

Amos Azaria
Data Science Center
Computer Science Department
Ariel University, Israel
amos.azaria@ariel.ac.il

Noam Hazon
Computer Science Department
Ariel University, Israel
noamh@ariel.ac.il

ABSTRACT

The computational problem of Influence maximization concerns the selection of an initial set of nodes in a social network such that, by sending this set a certain message, its exposure through the network will be the highest. We propose to study this problem from a utilitarian point of view. That is, we study a model where there are two types of messages; one that is more likely to be propagated but gives a lower utility per user obtaining this message, and another that is less likely to be propagated but gives a higher utility. In our model the utility from a user that receives both messages is not necessarily the sum of the two utilities. The goal is to maximize the overall utility.

Using an analysis based on bisubmodular functions, we show a greedy algorithm with a tight approximation ratio of $\frac{1}{2}$. We develop a dynamic programming based algorithm that is more suitable to our setting and show through extensive simulations that it outperforms the greedy algorithm.

KEYWORDS

Influence maximization, Social networks

ACM Reference Format:
Yael Sabato, Amos Azaria, and Noam Hazon. 2020. Viral vs. Effective: Utility Based Influence Maximization. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 9 pages.

1 INTRODUCTION

In recent years, social networks have surged in popularity, enabling people to share information easily and interact with each other. One of the main properties of social networks is the fast spread of messages; due to the multiple links of the networks, when a user receives a message, she may transfer it to a subset of her neighbors in the network, which may, in turn, transfer the message to their neighbors, and so on. This phenomenon was used by several stakeholders to promote their goods, agendas or ideas. For example, marketing companies advertise by social networks [19], social movements reach the public in order to get their support [6], and politicians run several social network pages [16, 18]. Naturally, the research in this field is thriving.

One natural problem that arises is the Influence Maximization (IM) problem. In this problem one needs to select an initial set of users (of a given size), to receive a message, such that the message will reach the largest number of users in the network. Most of the research on the IM problem concentrates on measuring a spread of a messages by counting the number of users that received that message, and this measure is used even where there are multiple types of messages. However, in many setting the effect of a message depends on its type, and thus counting the number of users that received any message is not the appropriate objective. For example, suppose there is an association that is promoting anti-smoking by running a campaign on a social network. Assume that there are two potential types of anti-smoking advertisements. One advertisement consists of humorous content[1], and thus it has a high potential to become viral and heavily spread throughout the network. However, this message has a low potential of affecting users to quit smoking. The second type of advertisement consists of harsh content[2], and it is thus more effective but less likely to spread. As another example, consider two advertisements for vaccine promotion. One advertisement includes graphic pictures of people who refused to vaccinate, and the other advertisement includes fun facts regarding the necessity of vaccination.

In this paper we propose to study the IM problem from a utilitarian perspective. That is, we study a model where there are two types of messages; one that is more likely to be propagated but gives a lower utility per user obtaining this message, and another that is less likely to be propagated but gives a higher utility. Clearly, whenever a user receives the same message multiple times it does not affect the resulting utility. However, when a user receives messages from different types, it is natural to assume that the resulting utility should be at least as high as the utility from each one of the messages. On the other hand, the utility in this case should not be higher than the sum of utilities. Overall, the Utility Based Influence Maximization (UBIM) problem is to select two initial sets of users (with a given total size), one for each type of message that is sent, such that the sum of the resulting utilities will be maximized.

We first show that a greedy method for UBIM is guaranteed to reach at least 50% of the optimal solution. The analysis is based on the maximization of monotone and bisubmodular functions [15], which is an extension of the traditional approach that maximizes a submodular function [8]. We also prove that the approximation ratio is tight. We note that the greedy algorithm has a drawback in our setting. Specifically, since there are two types of messages, if the greedy algorithm decides to send one type of message to a specific user, this commitment may harm its performance later on, as it may turn out that the other type of message is more valuable at later stages. We thus introduce our Efficient Table-based Algorithm for Bisubmodular functions (ETAB), which does not commit to a specific type of message at early stages.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

[1] See for example https://www.youtube.com/watch?v=IKbxMIWCto0
[2] See for example https://www.youtube.com/watch?v=AIyqcST29wQ

For the evaluation of our algorithms we use two graphs, a random graph and a graph based on a known social network (Digg). We compared the performance of ETAB with the greedy algorithm, its more efficient variant (Celf), and with additional baseline heuristics, in terms of achieved utility. Our results demonstrate that ETAB outperforms the other alternatives.

The contribution of this paper is threefold:

- We study the influence maximization problem from a utilitarian perspective. This is a natural extension, which allows to study the trade-off between sending viral and effective messages. To the best of our knowledge, this model has not been investigated yet.
- We prove that a greedy algorithm achieves a tight approximation ratio of $\frac{1}{2}$, since the function in our model is bisubmodular.
- We introduce an Efficient Table-based Algorithm for Bisubmodular functions (ETAB) and show through simulation that it outperforms the greedy algorithm, as well as additional baselines, on two different graphs.

2 RELATED WORK

The problem of influence maximization was introduced by Domingos and Richardson [7]. Motivated by application of marketing, they point out that a satisfied customer is likely to recommend the product to her friends, which raises the probability for them to buy the product as well. Naturally, a good marketing strategy will be to filter the users of a certain network in order to find the most influential users. Domingos and Richardson model the problem as a Markov random field, and provide heuristics for choosing costumers with a large overall effect.

The seminal work of Kempe et al. [8] is the first to analyze the influence maximization as a discrete optimization problem. They show that the IM problem is NP-hard, but it can be analyzed as a maximization of a *monotone* and *submodular* set function. This problem was studied by [14], and shown to admit an approximation of $1 - 1/e \approx 63\%$. Thus, the IM problem has the same approximation ratio. However, unlike the problem studied by [14], in the IM problem the underlying set function cannot be evaluated exactly (in polynomial time) but it can be approximated by sampling, and thus the approximation ratio is slightly lower. Since the work of Kempe et al., the IM problem received a high amount of attention, see the recent surveys [1, 11].

An important generalization of the IM problem, derives from the observation that many products are being promoted through a social network simultaneously. Some of them are complementary to each other and others are competitive. For example, buying a cellphone increases the probability of buying a cellphone case, but decreases the probability of buying a different type of cellphone. Therefore, there are several works that generalizes the IM problem such that there are multiple messages.

Specifically, Borodin et al. [3] discuss competitive IM. That is, they analyze several models where the goal is to maximize the spread of technology A while there is a different competitive technology B that is also spread in the network. They show that the greedy approach of [8] is applicable only for a subset of their models.

Datta et al. model viral marketing of multiple products [5]. They assume that the products are independent of each other. That is, a user's decision to buy a product is independent of her decision to buy other products. They also assume that multiple messages can be initially sent to each user, but the number of such messages is limited. The objective of their work is to maximize the *number* of products that the users buy, since all products are equally counted. Datta et al. show a greedy algorithm for their problem with a $1/3-$approximation ratio.

Narayanam and Nanavati study a model where cross-sell among products is possible [13]. That is, in their model when a user buys the first product it increases the probability that she will buy a second product. While they define a utility for the objective, it is a simple sum over the utilities from buying the products separately. In addition, they study the IM problem according to the linear threshold model while we study the IM problem according to the independent cascade model (see Section 3.1 for details).

In our work we show that a greedy algorithm obtains an approximation ration of $\frac{1}{2}$. This result is obtained by proving that the utility function in UBIM is monotone and bisubmodular. Ohsaka and Yoshida [15] were the first to show that a greedy algorithm on a $k-$submodular and monotone set function achieves an approximation ratio of $\frac{1}{2}$. They further show that their solution can be applied to the IM problem with multiple messages. Their objective function is to maximize the sum of users receiving these messages. Overall, none of these works extend the IM problem to the setting in which every message has its own (different) utility while a user receiving multiple messages might yield a value that is different from the sum of utilities.

Another related extension of the IM problem is the topic-aware influence maximization problem, which was introduced by Barbieri et al. [2] and Chen et al. [4]. In this model the influence between a pair of users may differ depending on the topic. Barbieri et al. introduce the model, and provide methods for learning the diffusion probabilities from data of past diffusion. Chen et al. study efficient algorithms for the IM problem in this setting. Note that the objective function is identical to that of the standard IM problem, and the existence of multiple topics affects only the probabilities of influences among the users.

A different perspective is studied by Li et al. [10]. They consider multiple types of messages, and use an agent-based modelling to study the diffusion process of the different influences.

3 PRELIMINARIES

3.1 The independent cascade model

The research on the IM problem has considered two main models of diffusion: the linear threshold model, and the independent cascade model. We focus on the independent cascade model and generalize it to our setting. The independent cascade model works as follows. The social network is represented by a directed graph $G = (V, E)$, where each user of the network is represented by a node and every connection between two users is an edge. The process of diffusion consists of a message that is propagated thorough the network. During this process each node can either become active or inactive, where an active node indicates that the associated user is influenced by the message. For every edge $u \rightarrow v \in E$, let $p^{uv} \in [0, 1]$ be the

probability of the influence of u on v. That is, p^{uv} is the probability for v to become active after she received the message from u. The process starts with an initial seed set $S \subseteq V$, such that all the nodes of S are initialized with the message, and thus they become active. The process then unfolds in discrete steps according to the following rule. Every node $v \in V$ that becomes active, activates each currently inactive neighbour w with probability p^{vw}. Moreover, if multiple neighbors of a vertex w try to activate it at the same time, their attempts are considered in an arbitrary order. v does not attempt to activate its neighbours again. The process runs until no more activations occur.

As shown by Kempe et al. [8], the diffusion process is equivalent to first selecting the participating edges according to their probabilities, (by a series of flipping coins with the corresponding probability) obtaining a graph of connections $G' = (V, E')$. Then, every node $v \in V$ of the graph that has a directed path starting from one of the nodes of the seed and ending at v is assumed to be active. We note that in G' all edges, E', have a fixed probability of 1.0.

3.2 Monotone and submodular set functions

We now provide the basic definitions of a bisubmodular function. This function is later used to model the utility in our setting. Let U be a set of n nodes. 3^U is the set of all possible tuples of two disjoint subsets of U. That is, $X = (X_1, X_2) \in 3^U$ if $X_1, X_2 \subseteq U$ and $X_1 \cap X_2 = \emptyset$. Note that every such tuple can be viewed as a vector of size n over $\{0, 1, 2\}$. Let $f : 3^U \to \mathbb{R}^+$ be a function that takes two disjoint subsets of U and outputs a non negative real number. For every tuple $X = (X_1, X_2) \in 3^U$ and $e \in U \setminus (X_1 \cup X_2)$, let $\Delta_{e,1}(X_1, X_2) = f(X_1 \cup \{e\}, X_2) - f(X_1, X_2)$. That is, the contribution of adding e to X_1, as measured by f. Similarly, let $\Delta_{e,2}(X_1, X_2) = f(X_1, X_2 \cup \{e\}) - f(X_1, X_2)$.

Definition 3.1. A function $f : 3^U \to \mathbb{R}^+$ is *monotone* if for every tuple $X = (X_1, X_2) \in 3^U$ and every $e \in U \setminus (X_1 \cup X_2)$, it holds that $\Delta_{e,1}(X_1, X_2) \geq 0$, and $\Delta_{e,2}(X_1, X_2) \geq 0$. That is, adding nodes does not reduce the value of f.

For completeness, we first provide the definition of a submodular function, which receives a single set as their input.

Definition 3.2. A function $f : 2^U \to \mathbb{R}^+$ is *submodular* if for every two subsets $X, Y \subseteq U$ it holds that:

$$f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$$

Definition 3.3. A function $f : 3^U \to \mathbb{R}^+$ is *bisubmodular* if for every two tuples $X = (X_1, X_2) \in 3^U$ and $Y = (Y_1, Y_2) \in 3^U$ it holds that:

$$f(X) + f(Y) \geq f(X \sqcup Y) + f(X \sqcap Y)$$

where,

- $f(X \sqcup Y) = f(X_1 \cup Y_1 \setminus (X_2 \cup Y_2), X_2 \cup Y_2 \setminus (X_1 \cup Y_1))$.
- $f(X \sqcap Y) = f(X_1 \cap Y_1, X_2 \cap Y_2)$.

An equivalent definition for bisubmodularity (see [17]), is when both properties of *orthant submodularity* and *pairwise monotonicity* hold.

Definition 3.4. A function $f : 3^U \to \mathbb{R}$ is *orthant submodular* if for every tuples $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ such that $X_1 \subseteq Y_1$ and $X_2 \subseteq Y_2$, and every item $e \in U \setminus (Y_1 \cup Y_2)$: $\Delta_{e,1}(X) \geq \Delta_{e,1}(Y)$ and similarly: $\Delta_{e,2}(X) \geq \Delta_{e,2}(Y)$.

Definition 3.5. A function $f : 3^U \to \mathbb{R}$ is *pairwise monotone* if for every tuple $X = (X_1, X_2)$ and every $e \in U \setminus (X_1 \cup X_2)$, $\Delta_{e,1}(X) + \Delta_{e,2}(X) \geq 0$.

Note that monotonicity implies pairwise monotonicity.

4 OUR MODEL

In our work we consider the diffusion process of two types of messages M_1 and M_2 in a social network. The social network is represented by a directed graph $G = (V, E)$, where each $v \in V$ represents a user in the social network, and each edge $u \to v \in E$ represents the influence of u on v. Each user can be activated by each of the two messages. Thus, there are four possible combinations for activeness; a user can be active with regard to only M_1, only M_2, active with both the messages or inactive. For every edge $u \to v$ we define $p_1^{uv}, p_2^{uv} \in [0, 1]$ to be the influence probabilities. p_1^{uv} is the probability that user u activates user v with regard to M_1. Similarly, p_2^{uv} is the influence probability with regard to M_2.

For the diffusion process we consider a generalization of the independent cascade model (see Section 3.1). Our process starts with two initial seed sets $S_1, S_2 \subseteq V$, $S_1 \cap S_2 = \phi$, such that all the nodes of S_1 are initialized with message M_1 and all the nodes of S_2 are initialized with message M_2. The process then unfolds in discrete steps according to the following rule. Let $i, j \in \{1, 2\}$, $i \neq j$. Every node v that became active with message M_i activates each neighbour w not activated yet with message M_i, with probability p_i^{vw}. In addition, if multiple neighbors of a vertex w try to activate it at the same time, their attempts are considered in an arbitrary order. v does not attempt to activate its neighbours with message m_i again. The process runs until no more activations are possible.

As mentioned in Section 3.1, the diffusion process of the IC model is equivalent to first selecting the participating edges according to their probabilities, then, every node of the graph that can be reached by a directed path from a node in the seed is considered active. For our model, we need to bring to consideration the fact that each edge has two different diffusion parameters. For the selection of the participating edges we use a series of $2|E|$ coin flips π, (Two for each edge), obtaining a multi-graph $G_\pi = (V, E_1 \cup E_2)$ such that the edges of E_1 and E_2 represents the connections of the nodes regarding the messages M_1 and M_2 respectively. Every node $v \in V$ that can be reached by a directed path from a node in S_1 is assumed to be active with regard to M_1. (Note that all the edges of this directed path should be from E_1). Similarly, every node $v \in V$ that can be reached by a directed path from a node in S_2 is assumed to be active with regard to M_2. (Again, all the edges of this directed path are from E_2). We note that in G_π all edges $e \in E_1$, have a fixed probability $p_1^e = 1.0$ and all edges $e \in E_2$, have a fixed probability $p_2^e = 1.0$.

Given a siries of coinflips π and a graph G_π, Let $A_1(S_1)$ be the set of all nodes that are active with M_1 at the end of the process, when S_1 is the seed set for M_1. Similarly, Let $A_2(S_2)$ be the set of all nodes that are active with M_2 at the end of the process, when S_2 is the seed set for M_2. Note that $A_1(S_1)$ depends only on the set S_1 and $A_2(S_2)$ depends only on the set S_2. Further note that $A_1(S_1) \cap A_2(S_2)$ consist of nodes that are active with both messages, $A_1(S_1) \setminus A_2(S_2)$

consist of nodes that are active with M_1 and are inactive with M_2, and $A_2(S_2) \setminus A_1(S_1)$ consist of nodes that are active with M_2 and are inactive with M_1.

We assume that each active node is associated with some utility. Let $u_1 \geq 0$ be the utility for nodes that are active only with M_1. Similarly, let $u_2 \geq 0$ be the utility for each node that is active only with M_2, and let $u_{1,2}$ be the utility for nodes that are active with both messages.

We define the utility function $\sigma_\pi(S_1, S_2)$ to be the sum of every active node at the end of the process multiplied with the corresponding utility, i.e,

$$\sigma_\pi(S_1, S_2) = u_1 \cdot |A_1 \setminus A_2| + u_2 \cdot |A_2 \setminus A_1| + u_{1,2} \cdot |A_1 \cap A_2|$$

Where we abbreviate $A_1(S_1)$ and $A_2(S_2)$ to A_1 and A_2 respectively. Finally, the overall utility function is the weighted average of each $\sigma_\pi(S_1, S_2)$ multiplied by the probability for π to occur, $p(\pi)$:

$$\sigma(S_1, S_2) = \sum_\pi \sigma_\pi(S_1, S_2) \cdot p(\pi)$$

Definition 4.1 (UBIM). Given an integer B (the budget), the *Utility Based Influence Maximization problem* is to find seed sets $X = (S_1, S_2)$ such that $|S_1| + |S_2| \leq B$, that maximize the utility $\sigma(S_1, S_2)$.

5 ALGORITHMS

5.1 The greedy approach

We extend the greedy algorithm that was presented by Kempe et al. to the UBIM problem. It runs as follows. First, two sets $S_1 = \emptyset, S_2 = \emptyset$ are initialized. At any step $1 \leq i \leq B$ the algorithm greedily chooses a node $e \in V \setminus (S_1 \cup S_2)$ and a message type $i \in \{1, 2\}$ such that adding e to S_i gives the largest marginal gain (see Algorithm 1).

Algorithm 1: Greedy

Input: A directed graph $G = (V, E)$ and an integer B.
Output: Two sets $S_1, S_2 \in V$, with $|S_1| + |S_2| = B$.
Initiate $S_1 \leftarrow \emptyset$, $S_2 \leftarrow \emptyset$;
for $i = 1$ *to* B **do**
  $u' \leftarrow \arg\max_{u \in V \setminus (S_1 \cup S_2)} \{\Delta_{e,1}(S_1, S_2)\}$;
  $u'' \leftarrow \arg\max_{u \in V \setminus (S_1 \cup S_2)} \{\Delta_{e,2}(S_1, S_2)\}$;
  if $u' \geq u''$ **then**
    $S_1 \leftarrow S_1 \cup \{u'\}$;
  else
    $S_2 \leftarrow S_2 \cup \{u''\}$;
return (S_1, S_2).

In this section, we show that if $\max(u_1, u_2) \leq u_{1,2} \leq u_1 + u_2$, then our utility function $\sigma(\cdot, \cdot)$, is monotone and bisubmodular, and therefore the greedy algorithm guarantees at least 50% of the optimal solution.

Theorem 5.1. *If* $\max(u_1, u_2) \leq u_{1,2} \leq u_1 + u_2$ *then the function* $\sigma(\cdot, \cdot)$ *is monotone and bisubmodular.*

Proof. We start with monotonicity. We need to show that for every tuple $X = (X_1, X_2) \in 3^U$, and every node $e \in U \setminus (X_1 \cup X_2)$,

$$\sigma(X_1 \cup \{e\}, X_2) \geq \sigma(X_1, X_2). \quad (1)$$

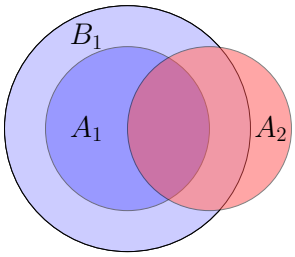

Figure 1: Venn's diagram with 3 sets, such that $A_1 \subseteq B_1$

$$\sigma(X_1, X_2 \cup \{e\}) \geq \sigma(X_1, X_2). \quad (2)$$

We will show (1) by:

$$\sigma(X_1 \cup \{e\}, X_2) - \sigma(X_1, X_2) \geq 0.$$

For (2) the analysis is symmetric. For given series of coin flip π and a corresponding graph G_π, Fix a tuple $X = (X_1, X_2)$ and a vertex $e \in U \setminus (X_1 \cup X_2)$, Let $A_1 = A_1(X_1)$, $A_2 = A_2(X_2)$ and let $B_1 = A_1(X_1 \cup \{e\})$. Note that $A_1 \subseteq B_1$ (see Figure 1). We get:

$$\sigma_\pi(X_1, X_2) = u_1 \cdot |A_1 \setminus A_2| + u_2 \cdot |A_2 \setminus A_1| + u_{1,2} \cdot |A_1 \cap A_2|$$

After adding e to S_1, we have:

$$\sigma_\pi(X_1 \cup \{e\}, X_2) = u_1 \cdot |B_1 \setminus A_2| + u_2 \cdot |A_2 \setminus B_1| + u_{1,2} \cdot |B_1 \cap A_2|$$

Let $\Delta_{\pi,e,1}(X) = \sigma_\pi(X_1 \cup \{e\}, X_2) - \sigma_\pi(X_1, X_2)$. Observe that in $B_1 \setminus (A_1 \cup A_2)$ there are nodes that where inactive for both messages and became active with M_1. Furthermore, in $(B_1 \cap A_2) \setminus A_1$ there are nodes that where active only with M_2 and became active with both messages. Therefore:

$$\Delta_{\pi,e,1}(X) = u_1 \cdot |B_1 \setminus (A_1 \cup A_2)| + (u_{1,2} - u_2) \cdot |(B_1 \cap A_2) \setminus A_1| \geq 0$$

where the inequality is due to the assumption that $u_{1,2} \geq \max(u_1, u_2)$, and that the sizes of the mentioned sets are non negative. We showed that the for every π, the function $\sigma_\pi(\cdot, \cdot)$ is monotone. Now, since $\sigma(\cdot, \cdot)$ is a convex combination of monotone functions, it is monotone as well.

For the bisubmodularity; we need to show that the properties of *orthant submodularity* and *pairwise monotonicity* hold (See Definitions 3.4 and 3.5 in Section 3.2). Since we showed monotonicity, pairwise monotonicity stems. For the orthant submodularity, Let π be a set of $2|E|$ coin flips, and let $G_\pi = (V, E_1 \cup E_2)$ be the corresponding multigraph. Let $X = (X_1, X_2) \in 3^U$ and $Y = (Y_1, Y_2) \in 3^U$ such that $X_1 \subseteq Y_1$ and $X_2 \subseteq Y_2$. Let $A_1 = A_1(X_1)$, $A_2 = A_2(X_2)$, $B_1 = A_1(Y_1)$ and $B_2 = A_2(Y_2)$. note that $A_1 \subseteq B_1$ and $A_2 \subseteq B_2$. In addition, Let $e \in U \setminus (Y_1 \cup Y_2)$. We will prove the claim for the case of adding e to X_1. The corresponding case (of adding e to X_2) is symmetric.

We label the nine[3] different areas of those four sets as follows (see Figure 2 for a Venn diagram):

- $C_1 = U \setminus (B_1 \cup B_2)$
- $C_2 = B_1 \setminus (B_2 \cup A_1)$
- $C_3 = A_1 \setminus B_2$
- $C_4 = (B_2 \cap A_1) \setminus A_2$
- $C_5 = A_1 \cap A_2$
- $C_6 = (B_1 \cap A_2) \setminus A_1$
- $C_7 = (B_1 \cap B_2) \setminus (A_1 \cup A_2)$
- $C_8 = A_2 \setminus B_1$
- $C_9 = B_2 \setminus (A_2 \cup B_1)$

[3]Note that a four set Venn diagram has $2^4 = 16$ different areas, here since $A_1 \subseteq B_1$ and $A_2 \subseteq B_2$ seven of thos areas are empty.

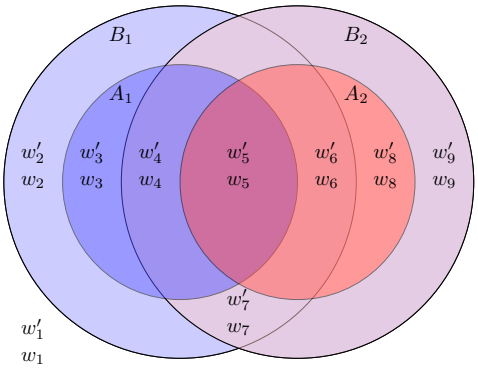

Figure 2: Venn's diagram with 4 sets such that $A_1 \subseteq B_1$ and $A_2 \subseteq B_2$

For every $1 \leq k \leq 9$, let W'_k be the set of nodes in C_k that are affected by e with regard to M_1. In other words; for every node $w \in W'_k$, G_π contains a directed path from e to w. (Note that all the path edges are from E_1). Moreover, let W_k be the set of nodes in C_k that are not affected by e. We denote be w'_k and w_k the number of nodes in W'_k and W_k respectively.

We need to show that $\Delta_{e,1}\sigma(X) - \Delta_{e,1}\sigma(Y) \geq 0$. Since we focus on the graph G_π, let $\Delta_{\pi,e,1}(X) = \sigma_\pi(X_1 \cup \{e\}, X_2) - \sigma_\pi(X_1, X_2)$ and Let $\Delta_{\pi,e,1}(Y) = \sigma_\pi(Y_1 \cup \{e\}, Y_2) - \sigma_\pi(Y_1, Y_2)$. We first note that:

$$\sigma_\pi(X_1, X_2) = u_1 \cdot (w_3 + w'_3 + w_4 + w'_4) + u_2 \cdot (w_6 + w'_6 + w_8 + w'_8) + u_{1,2} \cdot (w_5 + w'_5)$$

After adding e to X_1, the nodes of W'_1, W'_2, W'_7 and W'_9, which where inactive, became active with M_1, the nodes of W'_3, W'_4 and W'_5 did not change, and the nodes of W'_6 and W'_8, which where active only with M_2, became active with both messages. Thus:

$$\sigma_\pi(X_1 \cup \{e\}, X_2) = u_1 \cdot (w_3 + w'_3 + w_4 + w'_4 + w'_1 + w'_2 + w'_7 + w'_9) + u_2 \cdot (w_6 + w_8) + u_{1,2} \cdot (w_5 + w'_5 + w'_8 + w'_6)$$

We get,

$$\Delta_{\pi,e,1}\sigma(X) = u_1 \cdot (w'_1 + w'_2 + w'_7 + w'_9) - u_2 \cdot (w'_6 + w'_8) + u_{1,2} \cdot (w'_6 + w'_8) \quad (3)$$

Similarly, the analysis for $\Delta_{\pi,e,1}\sigma(Y)$ gives:

$$\sigma_\pi(Y_1, Y_2) = u_1 \cdot (w_2 + w'_2 + w_3 + w'_3) + u_2 \cdot (w_8 + w'_8 + w_9 + w'_9) + u_{1,2} \cdot (w_4 + w'_4 + w_5 + w'_5 + w_6 + w'_6 + w_7 + w'_7)$$

After adding e to Y_1, all nodes in W'_1, which were inactive, became active with M_1. the nodes in W'_8 and W'_6, which were active only with M_2, became active with both messages. This gives us:

$$\sigma_\pi(Y_1 \cup \{e\}, Y_2) = u_1 \cdot (w_3 + w'_3 + w_2 + w'_2 + w'_1) + u_2 \cdot (w_8 + w_9) + u_{1,2} \cdot (w_4 + w'_4 + w_5 + w'_5 + w_6 + w'_6 + w_7 + w'_7 + w'_8 + w'_9)$$

Therefore:

$$\Delta_{\pi,e,1}\sigma(Y) = u_1 \cdot (w'_1) - u_2 \cdot (w'_8 + w'_9) + u_{1,2} \cdot (w'_8 + w'_9) \quad (4)$$

Finally, let us now show that (3)–(4)≥ 0:

$$\Delta_{\pi,e,1}\sigma(X) - \Delta_{\pi,e,1}\sigma(Y) =$$
$$u_1 \cdot (w'_2 + w'_7 + w'_9) - u_2 \cdot (w'_6) + u_2(w'_9) + u_{1,2} \cdot (w'_6) - u_{1,2} \cdot (w'_9) =$$
$$u_1 \cdot (w'_2 + w'_7) + (-u_2 + u_{1,2}) \cdot (w'_6) + (u_1 + u_2 - u_{1,2}) \cdot (w'_9) \geq 0$$

where the inequality is due to assumption that $\max(u_1, u_2) \leq u_{1,2} \leq u_1 + u_2$, and that all the w's are non negative integers. We get that $\sigma_\pi(\cdot,\cdot)$ is bisubmodular. Finally, $\sigma(\cdot,\cdot)$ is a convex combination of bisubmodular functions, thus it is also a bisubmodular function. □

An important improvement to the time complexity of the greedy algorithm is the Celf algorithm, which is an extension of the algorithm introduced by [12] and [9] to UBIM. For each node the algorithm calculates a tuple $T = (n, a_1, a_2, valid)$, such that $T[n]$ is the node name, $T(a_1), T(a_2) \in \mathbb{R}$ are the expected addition to the overall utility when adding n to S_1 or S_2 respectively, and $T(valid) \in \mathbb{N}$ is the index of the iteration of the last evaluation of a_1 and a_2. The algorithm uses a queue that sorts the tuples by the value $\max(a_1, a_2)$ of each tuple. At every iteration, a tuple T is pulled from the queue; if its valid value is equal to the current iteration, then $T(n)$ is added to S_1 or S_2 (according to $\max(a_1, a_2)$). Otherwise, u's utility is sampled again, $T(a_1)$, $T(a_2)$ and $T(valid)$ are updated and T is inserted back into the queue.

5.2 The dynamic programming approach

One major drawback of the greedy algorithm (and Celf) is that, since there are two types of messages, if it decides to send one type of message to a specific user, this commitment may harm its performance later on, as it may turn out that the other type of message is more valuable at later stages. For example, consider Figure 3. The red arrows correspond to message 1, and the blue arrows correspond to message 2. Note that the influence probability is 1.0. For this example assume that $u_1 = 1$, $u_2 = 1.5$, and $u_{1,2} = 1.5$, and that $B = 2$. Clearly, at the first stage the greedy algorithm will select node A and add it to S_1; this will result in a utility of 5. In the second step, the greedy algorithm will select node B and add it to S_2; this will result in an additional utility of 1.5, thus resulting in a total utility of 6.5. However, an algorithm that would not commit to using the message 1 in the first step, could add nodes B and C to S_2 and yield a utility of 7.5. To overcome this problem, we introduce

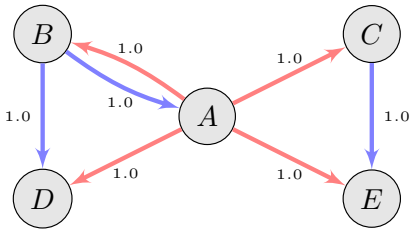

Figure 3: A graph demonstrating the problem with the greedy algorithm committing to a specific message type at early stages.

our Efficient Table-based Algorithm for Bisubmodular functions

(ETAB), which does not commit to a specific type of message at early stages. We first describe a dynamic programming based algorithm (TAB), and then describe a method for using insights from Celf to create the Efficient version of TAB (i.e. ETAB). In order not to commit to a specific type of message TAB builds its solution in two dimensions.

TAB uses the upper triangle of a table Mat of size $(B+1)\times(B+1)$. Each cell of Mat consists of a tuple $T=(S_1,S_2)$, such that $T(S_1), T(S_2)\subseteq V$ are disjoint seed sets. In each (i,j) cell of the table, it holds that $|S_1|=i$ and $|S_2|=j$. TAB starts with the tuple $T=(\emptyset,\emptyset)$ in cell $(0,0)$. The algorithm then fills in the first row; for every $1\leq i\leq B$, let $R_{i-1}=(R_1,\emptyset)$ be the tuple in cell $(i-1,0)$. The node $n\in V\setminus R_1$, that maximizes the value of the marginal gain to $\sigma(R_1,\emptyset)$, (i.e. the output of $\max_{n\in V\setminus R_1}(\Delta_{n,1}(R_1,\emptyset))$) is selected. Next the tuple $R_i=(R_1\cup\{n\},0)$ is stored in cell $(i,0)$. Similarly, TAB fills in the first column; for every $1\leq i\leq B$, let $C_{i-1}=(\emptyset,C_2)$ be the tuple in cell $(0,i-1)$. The node $n\in V\setminus C_2$, that maximizes the value of the marginal gain to $\sigma(\emptyset,C_2)$ is selected, and the tuple $C_i=(0,C_2\cup\{n\})$ is stored in cell $(0,i)$. The next phase is to fill in the cells of the middle; For each cell (k,l) TAB builds two optional tuples; T' is created by greedily adding an unselected node to seed 1 of the tuple in cell $(k-1,l)$. Similarly, T'' is created by greedily adding an unselected node to seed 2 of the tuple in cell $(k,l-1)$. TAB Then picks the tuple with the higher expected utility and store it in the cell (k,l).

After filling up all the cells of the table with indexes (i,j) such that $i+j\leq B$, TAB outputs the tuple with the maximal expected utility over the diagonal cells of the table (where $i+j=B$). see Algorithm 2.

Reconsidering the example of the graph in Figure 3. We show in Table 1 the outcome of applying TAB on those settings. TAB will output the maximum utility of the tuples in the diagonal (light blue) cells. Thus, the output utility of TAB is 7.5.

We believe that the approximation ratio of the TAB algorithm is very close to 50%. Moreover, this algorithm cannot guarantee an approximation ratio that is higher than 50%.

Lemma 5.2. *The approximation ratio of the TAB algorithm is at most* $50\%+\varepsilon$. *In other words, there is a case in which the value of the solution of TAB is less than* $50\%+\varepsilon$ *of the optimal solution. I.e.:* $f(S_{TAB})\leq f(O)+\varepsilon$.

Proof. We present an example where TAB outputs a solution that is just slightly higher then 50% of the optimal value. Let f be a monotone and bisubmodular set function. Let $U=\{1,2\}$, the function values are represented in Table 2. Since we assume that f is monotone and bisubmodular we get the following inequalities:

- $\max(x_1,x_2)\leq z_1\leq x_1+x_2$,
- $\max(x_1,y_2)\leq z_2\leq x_1+y_2$,
- $\max(x_2,y_1)\leq z_3\leq x_2+y_1$, and
- $\max(y_1,y_2)\leq z_4\leq y_1+y_2$.

Note that the lower bounds are due to the monotonicity of f and the upper bounds are due to the bisubmodularity of f.

Let $p\in\mathbb{R}^+$ and let $\varepsilon>0$. We further assume that $x_1=p+\varepsilon$, $x_2=p+2\varepsilon/3$, $y_1=p$ and $y_2=p-\varepsilon/3$. In addition, $z_3=2p+2\varepsilon/3$, $z_1=z_2=p+\varepsilon$ and $z_4=p+\varepsilon/3$. Note that $z_3\geq max(z_1,z_2,z_4)$, i.e. z_3 is the optimal solution when $B=2$. Table 3 shows the outcome of running TAB on the settings of f with $B=2$:

As we described the algorithm, TAB will output the tuple that corresponds to $\max(z_1,z_2,z_4)$, Thus the output will be $z_1=p+\varepsilon$. On the other hand, 50% of the optimal solution is $\frac{1}{2}z_3=\frac{1}{2}(2p+2\varepsilon/3)=$

Algorithm 2: TAB

```
Mat ← a (B+1)×(B+1) matrix of tuples;
Mat[0][0] ← (∅, ∅);
for i = 1 to B do
    S1 ← Mat[i−1][0](S1);
    n ← arg max_{n∈V\S1}(Δ_{n,1}(S1, ∅));
    Mat[i][0] ← (S1 ∪ {n}, ∅);
for i = 1 to B do
    S2 ← Mat[0][i−1](S2);
    n ← arg max_{n∈V\S2}(Δ_{n,2}(∅, S2));
    Mat[0][i] ← (∅, S2 ∪ {n});
for i = 1 to B do
    for j = 1 to (B−i) do
        L1 ← Mat[i−1][j](S1);
        L2 ← Mat[i−1][j](S2);
        l ← arg max_{n∈V\(L1∪L2)}(Δ_{n,1}(L1, L2));
        U1 ← Mat[i][j−1](S1);
        U2 ← Mat[i][j−1](S2);
        u ← arg max_{n∈V\(U1∪U2)}(Δ_{n,2}(U1, U2));
        if σ(L1 ∪ {l}, L2) > σ(U1, U2 ∪ {u}) then
            Mat[i][j] ← (L1 ∪ {l}, L2);
        else
            Mat[i][j] ← (U1, U2 ∪ {u});
(i, j) ← arg max_{i+j=B} (σ(Mat[i][j]));
return Mat[i][j];
```

$\lvert S_1\rvert$ \ $\lvert S_2\rvert$	0	1	2
0	0 $(\emptyset,\emptyset)$	5 $(\{A\},\emptyset)$	5 $(\{A,C\},\emptyset)$
1	4.5 $(\emptyset,\{B\})$	6.5 $(\{A\},\{B\})$	–
2	7.5 $(\emptyset,\{B,C\})$	–	–

Table 1: The table that TAB builds when running on the settings of Figure 3. TAB's output is $(\emptyset,\{B,C\})$, and the corresponding utility value is 7.5.

$B=0$	$f(\emptyset,\emptyset)=0$	
$B=1$	$f(\{1\},\emptyset)=x_1$	$f(\emptyset,\{1\})=y_1$
	$f(\{2\},\emptyset)=x_2$	$f(\emptyset,\{2\})=y_2$
$B=2$	$f(\{1,2\},\emptyset)=z_1$	$f(\{2\},\{1\})=z_3$
	$f(\{1\},\{2\})=z_2$	$f(\emptyset,\{1,2\})=z_4$

Table 2: A monotone and bisubmodular function for the proof of Lemma 5.2.

$p + \varepsilon/3$. Indeed, if we add ε to half the optimal solution we get $p + 4\varepsilon/3$, which is greater than z_1. □

$\lvert S_1\rvert$ \ $\lvert S_2\rvert$	0	1	2
0	0 $(\emptyset, \emptyset)$	$x_1 = p + \varepsilon$ $(\{1\}, \emptyset)$	$z_1 = p + \varepsilon$ $(\{1, 2\}, \emptyset)$
1	$y_2 = p + \varepsilon/3$ $(\emptyset, \{2\})$	$z_2 = p + \varepsilon$ $(\{1\}, \{2\})$	–
2	$z_4 = p + \varepsilon/3$ $(\emptyset, \{1, 2\})$	–	–

Table 3: The table that TAB builds when running on the settings of the function given in Table 2.

Corollary 5.3. *The approximation ratio of the greedy algorithm is at most* $50\% + \varepsilon$. *I.e. the approximation ratio of the greedy algorithm is tight.*

Proof. Observe that if the greedy algorithm is applied to the settings of the function in the proof of Lemma 5.2, the greedy algorithm will output the value of z_1 or z_2 which is $p + \varepsilon$, while the optimum value is $2p + 2\varepsilon/3$. □

Unfortunately, TAB is computationally expensive; However, as in the greedy algorithm, we may use the queue method to order the nodes and reduce the amount of evaluations. This brings us to our final algorithm, the Efficient version of TAB (i.e. ETAB). The general idea is similar to that of TAB except that every tuple contains additional two queues Q_1 and Q_2. Each queue take the unselected nodes of each cell and sorts them in decreasing order of the expected contribution they might add to the overall utility (see Algorithm 3).

Like in TAB, ETAB uses the upper triangle of a table Mat of size $(B + 1) \times (B + 1)$. Each cell of Mat consists of a tuple $T = (S_1, S_2, u, Q_1, Q_2)$, such that $T(S_1), T(S_2) \subseteq V$ are disjoint seed sets. $T(u) \in \mathbb{R}^+$ is the expected utility $\sigma(T(S_1), T(S_2))$. Moreover, $T(Q_1)$ and $T(Q_2)$ are two priority queues that sort all the nodes in $V \setminus (T(S_1) \cup T(S_2))$ in decreasing order of their expected contributions $\Delta_{T(n),1}(T(S_2), T(S_2))$ and $\Delta_{T(n),2}(T(S_2), T(S_2))$, respectively. In addition, for each (i, j) cell of the table, it holds that $|S_1| = i$ and $|S_2| = j$. ETAB runs similarly to TAB, except that when ever it needs to select a node in order to add it to a certain seed of a certain cell, it uses the cell's priority queue. Thus reducing the amount of the function evaluations (see Algorithm 3).

We note that the space complexity of ETAB is very high since we store many queues (each queue has space complexity of $O(n)$). Each cell has two queues and we use $(B + 1)^2/2 = O(B^2)$ cells. To slightly reduce space complexity, after calculating the tuples T' and T'' of the last loop, we can free the queues of the *above* cell, as we do not use them again. Thus, the number of queues we store during the running of the algorithm is at most $2(2B + 1) = O(B)$, therefore the overall space complexity of ETAB is $O(Bn)$.

Algorithm 3: ETAB (Efficient Table-based Algorithm for Bisubmodular functions)

```
Mat ← a (B + 1) × (B + 1) matrix of queue tuples.
Q1 ← a priority order queue of nodes, sorted by Δ_{n,1}(∅, ∅)
Q2 ← a priority order queue of nodes, sorted by Δ_{n,2}(∅, ∅)
Mat[0][0] ← (∅, ∅, 0, Q1, Q2)
for i = 1 to B do
    Mat[i][0] ← copy Mat[i − 1][0]
    Add a node to Mat[i][0](S1) by using the queue Mat[i][0](Q1)
    Update Mat[i][0](Q1), Mat[i][0](Q2) and Mat[i][0](u)
for i = 1 to B do
    Mat[0][i] ← copy Mat[0][i − 1]
    Add a node to Mat[0][i](S2) by using the queue Mat[0][i](Q2)
    Update Mat[0][i](Q1), Mat[0][i](Q2) and Mat[0][i](u)
for i = 1 to B do
    for j = 1 to B − i do
        T' ← copy Mat[i − 1][j]
        Add a node to T'(S1) by using the queue T'(Q1)
        Update T'(Q1), T'(Q2) and T'(u)
        T'' ← copy Mat[i][j − 1]
        Add a node to T''(S2) by using the queue T''(Q2)
        Update T''(Q1), T''(Q2) and T''(u)
        if T'(u) > T''(u) then
            Mat[i][j] ← T'
        else
            Mat[i][j] ← T''
(i, j) ← arg max_{i+j=B}(Mat[i][j](u))
return (Mat[i][j](S1), Mat[i][j](S2))
```

6 EXPERIMENTS

For evaluating the performance of ETAB and comparing it to the greedy algorithm, Celf and additional baselines, we built two graphs that include diffusion parameters on their edges. Our first graph is a randomly induced graph, with 1,000 nodes and a density of 0.001. That is, each directed edge was inserted to the graph with a probability of 0.001. Then, for every edge $u \to v$, we assigned $p_1^{uv} = 0.5/degree_{in}(v)$ and $p_2^{uv} = 1/degree_{in}(v)$. The average values of p_1 and p_2 were 0.3184 and 0.6367, respectively.

Our second graph is based on the Digg2009 data-set[4], a publicly available data-set that was obtained from the Digg website. Digg is a social news website, that allows people to vote on web content. The data-set contains friendship connections between the users, and a list of votes for various stories; each vote contains a story id, a vote date and user id. In order to extract the diffusion parameters we considered that a vote of user v is influenced by her friend u, if v voted for a story after u has voted for it. For every edge $u \to v$, we calculated a parameter x, which is the number of stories that v voted for as an influence of u, divided by the overall number of stories that u voted for. We then assigned $p_2^{uv} = x$, and $p_1^{uv} = 0.5 \cdot x$. The average values of p_1 and p_2 were 0.05832 and 0.11664, respectively. We deleted edges with both $p_1 = 0$ and $p_2 = 0$ and removed all

[4] https://www.isi.edu/~lerman/downloads/digg2009.html

isolated nodes. At the end of this process we randomly selected 1,000 nodes, resulting with 51,398 edges.

In addition to ETAB, TAB, Greedy and Celf, we also ran the following baseline heuristics:

- **Degree**$_{count}$: The nodes are selected in descending order according to their degrees, and each selected node is randomly assigned either to S_1 or to S_2.
- **Degree**$_{expected}$: The nodes and the message types are selected in descending order according to the sums of their diffusion parameters. That is, for each node x, and for each type i, let $d_i^x = \sum_{xu \in E(g)} p_i^{xu}$; the nodes are selected according to $\max(d_1^x, d_2^x)$ and are added to the seed corresponding to the message type with the higher d_i^x.
- **Degree**$_{sampled}$ Every node n is assigned two values $SD_1(n) := \sigma(\{n\}, \emptyset)$ and $SD_2(n) := \sigma(\emptyset, \{n\})$. The nodes are selected in descending order of $\max(SD_1, SD_2)$, and each selected node is assign to S_1 if $SD_1 \geq SD_2$, or to S_2 if $SD_1 < SD_2$.
- **Random**: A random selection of nodes, which are randomly assigned to one of the seeds.

We assigned the following utilities $u_1 = 2$, $u_2 = 1$ and $u_{1,2} = 2.5$. The evaluations of the utility function was sampled 100 times during execution of each of the algorithms, and was sampled 10,000 times for final evaluation of the algorithm output. In order for our results to be less biased, we ran separate simulations for each Budget level; therefore, there may be a slight drop in the utility achieved by an algorithm when running it with some budget compared to the same algorithm when using a lower budget. The budget was set to multiplies of 10, up-to a budget of 200 initial nodes. Our results are presented in Figures 4 and 5. Note that some values are missing from the plot due to the extensive time required for computing them.

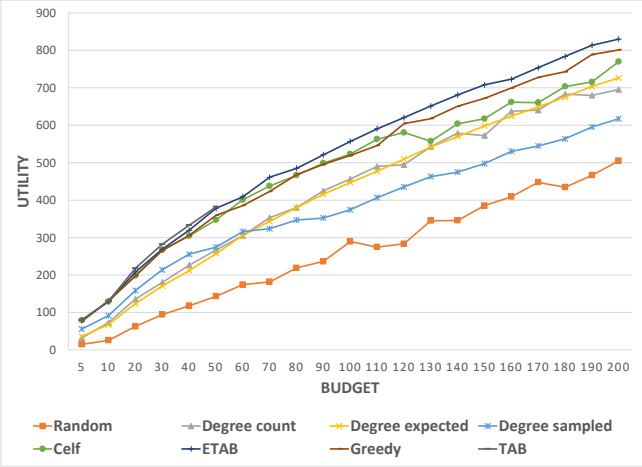

Figure 4: The utility of all the evaluated methods on the random graph.

As can be seen in Figures 4 and 5, ETAB achieves the highest utility (after TAB), outperforming all other methods at all budget levels. An interesting aspect, that has not been fully studied before, is that the greedy algorithm gives a better solution than Celf. We

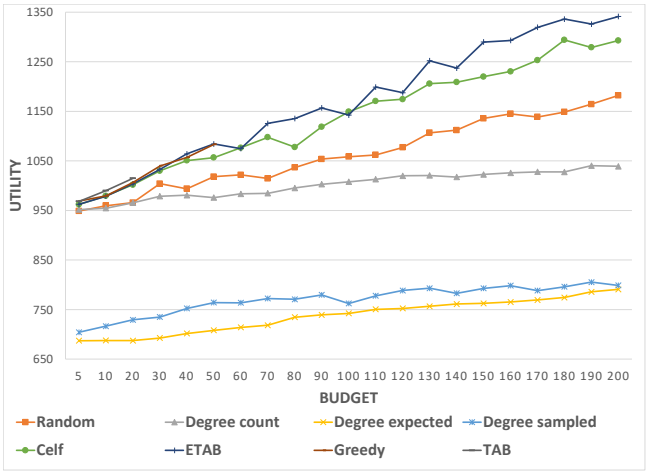

Figure 5: The utility of all the evaluated methods on the Digg based graph.

believe that this is because the stochastic nature of Celf, as it relies on samples to estimate the addition of each node to the overall utility. Since those samples are stochastic, there is an error rate that is accumulated during Celf's execution. Similar behavior is observed when comparing TAB with ETAB.

7 CONCLUSION & FUTURE WORK

In this paper we introduce the utility based influence maximization problem, which is a natural extension of the common influence maximization problem. We show that our utility function is monotone and bisubmodular and thus we provide a greedy algorithm that achieves an approximation ratio of $\frac{1}{2}$. We develop ETAB, which is a dynamic programming based algorithm suitable to our setting, and show that it outperforms the greedy algorithm.

We note that ETAB is a general solution, as it is applicable for any problem of maximization a monotone and bisubmodular function. For example, in the antennas placement problem one needs to decide where to place antennas such that the overall reception will be maximized. Similar to our setting, it is possible that there are two types of antennas: one that has a strong signal but a small bandwidth, and the other has a weaker signal but a larger bandwidth. In this setting the reception where there is an overlap between the antennas is assumed to be at least as high as the reception from each type of antenna, but no more than their sum. Therefore, ETAB can be used to efficiently place the two types of antennas.

For future work there are several interesting directions. First, we would like to apply ETAB to known problems in additional domains such as the antennas placement problem. In addition, we would like to extend our analysis to the setting where the utility function is not bisubmodular or non-monotone. Finally, our model can be extended to the case where different users yield different utilities form the same message.

ACKNOWLEDGMENTS

This research was supported in part by the Ministry of Science, Technology & Space, Israel.

REFERENCES

[1] Suman Banerjee, Mamata Jenamani, and Dilip Kumar Pratihar. 2018. A survey on influence maximization in a social network. *arXiv preprint arXiv:1808.05502* (2018).

[2] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2013. Topic-aware social influence propagation models. *Knowledge and information systems* 37, 3 (2013), 555–584.

[3] Allan Borodin, Yuval Filmus, and Joel Oren. 2010. Threshold models for competitive influence in social networks. In *International workshop on internet and network economics*. Springer, 539–550.

[4] Wei Chen, Tian Lin, and Cheng Yang. 2014. Efficient topic-aware influence maximization using preprocessing. *CoRR, abs/1403.0057* (2014).

[5] Samik Datta, Anirban Majumder, and Nisheeth Shrivastava. 2010. Viral marketing for multiple products. In *2010 IEEE International Conference on Data Mining*. IEEE, 118–127.

[6] Mario Diani. 2003. Social movements, contentious actions, and social networks:'From metaphor to substance'. *Social movements and networks: Relational approaches to collective action* (2003), 1–20.

[7] Pedro Domingos and Matt Richardson. 2001. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 57–66.

[8] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 137–146.

[9] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 420–429.

[10] Weihua Li, Quan Bai, Minjie Zhang, and Tung Doan Nguyen. 2018. Modelling multiple influences diffusion in on-line social networks. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1053–1061.

[11] Yuchen Li, Ju Fan, Yanhao Wang, and Kian-Lee Tan. 2018. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* 30, 10 (2018), 1852–1872.

[12] Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization techniques*. Springer, 234–243.

[13] Ramasuri Narayanam and Amit A Nanavati. 2012. Viral marketing for product cross-sell through social networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 581–596.

[14] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming* 14, 1 (1978), 265–294.

[15] Naoto Ohsaka and Yuichi Yoshida. 2015. Monotone k-submodular function maximization with size constraints. In *Advances in Neural Information Processing Systems*. 694–702.

[16] Sonja Utz. 2009. The (potential) benefits of campaigning via social network sites. *Journal of computer-mediated communication* 14, 2 (2009), 221–243.

[17] Justin Ward and Stanislav Živnỳ. 2016. Maximizing k-submodular functions and beyond. *ACM Transactions on Algorithms (TALG)* 12, 4 (2016), 47.

[18] David Wills and Stuart Reeves. 2009. Facebook as a political weapon: Information in social networks. *British Politics* 4, 2 (2009), 265–281.

[19] Wan-Shiou Yang, Jia-Ben Dia, Hung-Chi Cheng, and Hsing-Tzu Lin. 2006. Mining social networks for targeted advertising. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, Vol. 6. IEEE, 137a–137a.