Threshold Influence Model for Allocating Advertising Budgets

Atsushi Miyauchi MIYAUCHI.A.AA@M.TITECH.AC.JP
Graduate School of Decision Science and Technology, Tokyo Institute of Technology, Japan

Yuni Iwamasa YUNI_IWAMASA@MIST.I.U-TOKYO.AC.JP
Graduate School of Information Science and Technology, University of Tokyo, Japan

Takuro Fukunaga TAKURO@NII.AC.JP
National Institute of Informatics, JST, ERATO, Kawarabayashi Large Graph Project, Japan

Naonori Kakimura KAKIMURA@GLOBAL.C.U-TOKYO.AC.JP
Graduate School of Arts and Sciences, University of Tokyo, Japan

Abstract

We propose a new influence model for allocating budgets to advertising channels. Our model captures customer's sensitivity to advertisements as a threshold behavior; a customer is expected to be influenced if the influence he receives exceeds his threshold. Over the threshold model, we discuss two optimization problems. The first one is the budget-constrained influence maximization. We propose two greedy algorithms based on different strategies, and analyze the performance when the influence is submodular. We then introduce a new characteristic to measure the cost-effectiveness of a marketing campaign, that is, the proportion of the resulting influence to the cost spent. We design an almost linear-time approximation algorithm to maximize the cost-effectiveness. Furthermore, we design a better-approximation algorithm based on linear programming for a special case. We conduct thorough experiments to confirm that our algorithms outperform baseline algorithms.

1. Introduction

Recent development of online advertisement has opened up a lot of opportunities for companies to conduct a marketing campaign. They can promote their products through various online advertising channels such as search engine and social media, in addition to traditional media such as TV, newspapers, and radio. In marketing, one of the major decisions is how to allocate a given budget among such media channels in order to maximize the impact on a set of potential customers.

We may model as a bipartite graph in which one side is the set of possible marketing channels, and the other is the population of target customers. An edge between a channel i and a customer j indicates that i may influence j. For the above marketing problem, we can simply consider choosing a set of advertising channels to maximize the number of customers covered by the chosen channels. This is the classical combinatorial optimization problem called the *maximum coverage problem*. The problem is known to be NP-hard, but we can obtain an $(e/(e-1))$-approximate solution[1] using a greedy-type algorithm (Hochbaum, 1997). However, the maximum coverage problem only focuses on the number of covered customers, and cannot handle a probabilistic behavior of making influence in reality. Moreover, we do not consider allocating some amount of budgets to a channel.

Alon et al. (2012) proposed two fundamental marketing models, the *source-side influence model* and the *target-side influence model*, incorporating a probabilistic behavior of a market model as well as allowing us to allocate a budget to a source node. The two models focus on individual (and independent) interaction and influence between sources and targets. In their first model, each source determines a probability of their influence to targets, while each target does in the other model. The first model was extended to a more general setting (Soma et al., 2014).

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

[1] A feasible solution is α-*approximate* if its objective value is at least $1/\alpha$ times the optimal value. An α-*approximation algorithm* is one that returns an α-approximate solution for any instance.

1.1. Our Contribution

In this paper, we propose a threshold influence model over a bipartite graph. We capture customer's sensitivity to our campaign as a threshold behavior. In the proposed model, when we allocate some amount of budgets to a channel, the influence which a customer receives is defined by a (general) monotone function of the allocated budgets. Examples of a monotone function are those in both the Alon et al.'s influence models (2012). A customer is then influenced if the received influence exceeds his threshold. The detailed model definition will be presented in Section 2.

A threshold represents the different latent tendencies of customers. Such a model with thresholds has been studied in the context of viral marketing on social networks, particularly when an influence function is linear or submodular (see e.g., (Granovetter, 1978; Kempe et al., 2003; Mossel & Roch, 2010)). Our model extends these models so that we are allowed to allocate a budget to a source. Furthermore, in previous studies on viral marketing, a threshold is usually supposed to be uniformly random, as we do not know the customers' latent tendencies in advance. However, in a practical scenario over a bipartite market model, it is reasonable that we have observed their tendencies; some customers are easy to be influenced, while some are not. Thus we consider an influence model with specific thresholds to incorporate these aspects. Let us also remark that our model includes various influence maximization problems such as the Alon et al.'s target-side model, as well as clustering problems such as the densest k-subgraph problem (Feige et al., 2001).

Our main contribution is to discuss two optimization problems over the threshold influence model, which can be summarized as follows.

1. We consider finding a budget allocation that maximizes the number of influenced targets subject to a total budget constraint, which we refer to as the *maximum general-thresholds coverage problem*. We develop two simple algorithms based on different greedy strategies.

2. We show that when an influence to each target is determined by a monotone submodular function, the problem with random thresholds can be solved approximately within a factor of $e/(e-1)$.

3. We introduce a new characteristic to measure the performance of a marketing campaign, which we refer to as the *cost-effectiveness*. We propose an almost linear-time approximation algorithm to maximize the cost-effectiveness. Furthermore, we develop a linear-programming-based (LP-based) algorithm when a threshold function for each target is of a simple form.

4. The problem of maximizing the cost-effectiveness is closely related to clustering problems. Our proposed algorithms solve efficiently the weighted densest sub-hypergraph problem.

Let us describe our first and second results in details.

One of our proposed algorithms for the maximum general-thresholds coverage problem is a greedy algorithm, that repeatedly chooses the source with maximum marginal return. It is known that such an algorithm finds an $(e/(e-1))$-approximate solution for the maximum coverage problem, and moreover, this, together with some preprocessing, returns an $(e/(e-1))$-approximate solution for the source-side influence model. This is because the objective function is monotone and submodular (see Section 1.3 for the definition). In our model, the objective function is *not necessarily* submodular or supermodular. In fact, if a threshold is large, there is a case where adding one source makes no target influenced. In this case, we can choose any source, which seems hopeless to have good approximation.

However, we show that, if an influence to each target is determined by a monotone submodular function over integer lattice, the greedy strategy returns a well-approximate solution. In fact, when a threshold is given randomly, the expected value of the influence is monotone and submodular over integer lattice. Thus the expected influence can be maximized within a factor of $e/(e-1)$ in pseudo-polynomial-time, which was done by the greedy algorithm using the maximum expected marginal return (Soma et al., 2014). We here propose a slightly different greedy algorithm for the problem. The key idea is to design a general framework to reduce a monotone submodular function over integer lattice to a set function that is monotone and submodular. The proof is simple, and furthermore, our reduction framework provides approximation algorithms for problems related to a monotone submodular function over integer lattice, e.g., a multi-set generalization of (Iyer & Bilmes, 2013).

The greedy algorithm mentioned above has inefficiency in the sense that the algorithm requires $O(n^2)$ number of function evaluations even for the Boolean case, where n is the number of sources. It should be noted that, when an influence is submodular, we can reduce practical time complexity by the lazy-evaluation technique (Minoux, 1978), but it is impossible for a general case. We propose another greedy-type algorithm, which runs in almost linear time. In the algorithm, we start from the whole source set, and repeatedly remove the source node with the smallest contribution. The algorithm is a generalization of Asahiro et al. (2000) and Charikar (2000) for the densest subgraph problem. Note that the algorithm gives good approximation for the cost-effectiveness maximization as below. To evaluate the performance of the two proposed algorithms, we thoroughly conduct simulation on a real-world network

and artificially generated networks.

Let us describe our third and fourth results in details.

When we conduct a marketing campaign, it is often interesting to evaluate the performance of the campaign. The performance can be measured as the proportion of the resulting influence to the cost spent in the campaign, which we call the *cost-effectiveness*. Over our threshold model, we introduce the problem of maximizing the cost-effectiveness. Remark that the problem is useful to analyze a total budget in a marketing campaign; it tells us how much performance we could achieve (without a total budget constraint), which would help in readjusting a total budget to improve the performance.

For this problem, we first design an almost linear-time approximation algorithm. In fact, the second greedy algorithm in our first result achieves an approximation factor of the maximum degree of target nodes. In our context, customers usually communicate through a small number of media channels, and hence the factor is effective in practice. Furthermore, we develop an LP-based better-approximation algorithm when a threshold function for each target is of a simple form.

Our problem is closely related to clustering problems. In fact, the cost-effectiveness maximization includes the densest subgraph problem and the densest subhypergraph problem (see Section 2.2 for the definitions). These problems can be solved in polynomial time (Huang & Kahng, 1995), and approximately in linear time within a factor of the maximum size of hyperedges when hyperedges have the same size (Tsourakakis, 2014). Our results extend these ones so that we are allowed to have hyperedge weights and different hyperedge sizes.

Due to the space limitation, some proofs are omitted, which can be found in the Supplementary Material.

1.2. Related Work

Motivated by applications to marketing, Domingos and Richardson (2001) and Richardson and Domingos (2002) introduced *viral marketing*. They considered the problem of choosing a few "influential" members in a social network to influence a large fraction of the network through word-of-mouth effects. Such problem is called the *influence maximization*. There are two standard models, i.e., the *independent cascade* (IC) model (Goldenberg et al., 2001a;b) and the *linear threshold* (LT) model (Granovetter, 1978). Kempe et al. (2003) showed that for both the IC and LT models, the expected number of influenced nodes is a monotone and submodular function with respect to a seed set. This implies that an optimal solution can be efficiently approximated within a factor of $(e/(e-1)+\varepsilon)$ using a greedy-type algorithm (Nemhauser et al., 1978). Kempe et al. (2003) also proposed a commonly generalized model, called the *general threshold model*, which is proven to have submodularity (Mossel & Roch, 2010). Lu et al. (2011; 2012) introduced an LT model with deterministic thresholds.

The source-side influence model, introduced by Alon et al. (2012), focuses on independent influence between sources and targets, which can be viewed as a budgeted counterpart of the IC model over a bipartite graph. Our proposed model is a budgeted counterpart of the LT model, which also includes the target-side influence model (Alon et al., 2012) as a special case.

In addition to the budget allocation in marketing, our model has various applications in machine learning, which will be presented in Section 2 with detailed formulations.

1.3. Preliminaries

We conclude this section with definitions and notation necessary in this paper.

Let $G = (S, T; E)$ be a bipartite graph. For a subset $Y \subseteq T$, we denote by $\Gamma(Y)$ the set of nodes in S that are neighbors of some node in Y, i.e., $\Gamma(Y) = \{s \in S \mid \exists t \in Y, (s,t) \in E\}$. If Y has only one node t, we simply denote $\Gamma(t) = \Gamma(\{t\})$. We define $\Gamma(X)$ for $X \subseteq S$ similarly.

Let S be a finite set. We say that a set function $g : 2^S \to \mathbb{R}$ is *monotone* if $g(X) \leq g(Y)$ for all X, Y with $X \subseteq Y$. A set function g is *submodular* if it satisfies $g(X) + g(Y) \geq g(X \cup Y) + g(X \cap Y)$ for all $X, Y \subseteq S$. A set function g is *supermodular* if $-g$ is submodular. Intuitively, a submodular function demonstrates a "diminishing marginal return," that is, as we add an element to a larger set, the increase is smaller. Submodularity captures fundamental properties in viral marketing on social networks (Kempe et al., 2005; Mossel & Roch, 2010), and has various applications in machine learning (Krause & Golovin, 2014; Lin & Bilmes, 2011; Singh et al., 2012).

Let $\mathbb{Z}_+^S$ be the set of non-negative integer vectors in which each component is indexed by an element in a set S. A function $f : \mathbb{Z}_+^S \to \mathbb{R}$ is *monotone* if $f(x) \leq f(y)$ for all x, y with $x \leq y$ (i.e., $x(s) \leq y(s)$ for each $s \in S$). We say that f is a *submodular function over integer lattice* $\mathbb{Z}_+^S$ when it satisfies $f(x) + f(y) \geq f(x \vee y) + f(x \wedge y)$ for all integer vectors x and y, where $x \vee y$ and $x \wedge y$, respectively, denote the coordinate-wise maxima and minima, i.e., $(x \vee y)(s) = \max\{x(s), y(s)\}$ and $(x \wedge y)(s) = \min\{x(s), y(s)\}$ for each $s \in S$. This definition generalizes submodularity of set functions because if the domain is restricted to the unit hypercube, then the submodularity can be identified with that of set functions. A submodular function over integer lattice is used to analyze and extend the Alon et al.'s model (Soma et al., 2014).

2. Problem Definition

In this section, we define our model formally, and present several examples included in the model.

2.1. Maximum General-Thresholds Coverage

Let $G = (S, T; E)$ be a bipartite graph, where S is a set of media channels, T is a set of target customers, and an edge set $E \subseteq S \times T$ represents relationship between channels and customers. We are given a unit cost c_s for each source $s \in S$, a weight w_t for each target $t \in T$, and a total budget B being a positive integer. When we choose a set X of nodes in S, the set X makes influence on target nodes. Specifically, the influence by a subset X in S is determined as follows. Each target $t \in T$ has a threshold θ_t and a monotone set function $f_t : 2^{\Gamma(t)} \to \mathbb{R}_+$ where $f_t(\emptyset) = 0$. A target t is *influenced by* X if $f_t(X_t) \geq \theta_t$, where we denote $X_t = X \cap \Gamma(t)$.

The *maximum general-thresholds coverage* problem is to find $X \subseteq S$ that maximizes

$$f_G(X) = \sum_{t \in T : f_t(X_t) \geq \theta_t} w_t$$

subject to a budget constraint $c(X) := \sum_{s \in X} c_s \leq B$. When the graph G is clear from the context, we simply denote f instead of f_G.

We also consider the *multi-set version* of the problem. In the setting, each source $s \in S$ additionally has a capacity u_s. We allocate to S a positive integer vector $x = (x_s)_{s \in S}$ that satisfies a budget constraint $\sum_{s \in S} c_s x_s \leq B$. The vector x makes influence to targets t in T, which is defined by a monotone function f_t over integer lattice $\mathbb{Z}_+^{\Gamma(t)}$ and a threshold θ_t. We then aim to maximize the total weight of nodes influenced:

$$f_G(x) = \sum_{t \in T : f_t(x_t) \geq \theta_t} w_t,$$

where x_t is the subvector of x whose indices are $\Gamma(t)$.

As described below, the problem includes various optimization problems.

Example 1 (Maximum coverage problem)**.** Suppose that the function f_t is given by

$$f_t(X) = |X| \tag{1}$$

and the threshold $\theta_t = 1$ for every $t \in T$. Assume that $c_s = 1$ for any $s \in S$ and $w_t = 1$ for any $t \in T$. Then the problem is the well-known *maximum coverage problem*. In this case, the objective function f_G is monotone and submodular, and thus we can approximate the optimal value within a factor of $e/(e-1)$ (Hochbaum, 1997).

Example 2 (Maximum thresholds coverage problem)**.** Suppose that f_t is given by (1), and θ_t is a positive integer between 1 and $|\Gamma(t)|$. Then, when $c_s = 1$ for any $s \in S$, the objective is to find a set X of B nodes in S that maximizes the total weight of nodes t in T covered at least θ_t times by the edges from X.

This problem is called the *maximum thresholds coverage problem*, which was introduced by Alon et al. (2012) to investigate the target-side influence model in Example 3 below. They showed that this problem, i.e., our problem, too, is hard to approximate within a factor of $\Omega(B^\epsilon)$ for some $\epsilon > 0$, assuming a certain hypothesis about the average-case hardness of random 3SAT. On the positive side, they proposed an $O(\log B)$-approximation algorithm for the multi-set case when u_s is infinity.

Example 3 (Target-side influence model)**.** Alon et al. (2012) introduced a target-side influence model. In the model, each target $t \in T$ has a sequence of probabilities $(p_1^t, \ldots, p_B^t)$. A subset X of S makes $|X_t|$ independent trials to each t according to the probability sequence. Specifically, the probability f_t of t being influenced is given by

$$f_t(X) = 1 - \prod_{i=1}^{|X_t|} (1 - p_i^t). \tag{2}$$

The objective is to maximize the expected number of influenced nodes, i.e., $\sum_{t \in T} f_t(X)$. It should be noted that the function f_t is monotone, but not necessarily submodular or supermodular. The problem can be polynomially reduced to the maximum thresholds coverage problem (Example 2). Furthermore, in our problem setting with thresholds, we can consider maximizing the total weight of targets t in T whose probability (2) is at least θ_t.

Example 4 (Source-side influence model with threshold)**.** Suppose that we are given a probability p_s for each source $s \in S$, and that f_t is a probability by independent trials from sources. That is, f_t is given by

$$f_t(X) = 1 - \prod_{s \in X_t} (1 - p_s). \tag{3}$$

In our problem setting, we can consider maximizing the total weight of targets t in T whose probability (3) is at least θ_t.

Example 5 (Densest k-subgraph problem)**.** The *densest k-subgraph problem* is the problem of finding a set of k vertices that maximizes the number of edges in the subgraph induced by the k vertices. The best known approximation algorithm achieves an approximation factor of $O(n^{1/3-\delta})$ for some small $\delta > 0$ (Feige et al., 2001). Bhaskara et al. (2010) proposed an $O(n^{1/4+\varepsilon})$-approximation algorithm that runs in $n^{O(1/\varepsilon)}$ time for any $\varepsilon > 0$. Recently, Papailiopoulos et al. (2014) designed a nearly linear-time al-

gorithm whose approximation factor depends on the eigenvalues of the adjacency matrix. This problem can be reduced to our problem, as described in Example 6 below.

Example 6 (Densest k-subhypergraph problem)**.** Let V be a finite set and $\mathcal{E}$ be a family of subsets of V. The pair $H = (V, \mathcal{E})$ is called a *hypergraph*. Note that, when the size of each element in $\mathcal{E}$ is exactly two, H forms a graph. For a subset $X \subseteq V$, we denote by $\mathcal{E}(X)$ the set of hyperedges in the subhypergraph induced by X. In the *densest k-subhypergraph problem*, given a hypergraph $H = (V, \mathcal{E})$, we find a subset $X \subseteq V$ of size k that maximizes $|\mathcal{E}(X)|$.

This problem can be reduced to our problem. Indeed, a hypergraph H can be naturally represented as a bipartite graph as follows. For $H = (V, \mathcal{E})$, we construct a bipartite graph $G = (V, \mathcal{E}; E)$, where $E = \{(v, e) \mid v \in e\}$. Each node $e \in \mathcal{E}$ in G has degree exactly $|e|$. We set $\theta_e = |e|$ for every $e \in \mathcal{E}$. Then, for a subset $X \subseteq V$, the set $\mathcal{E}(X)$ is equivalent to $\{e \in \mathcal{E} \mid |X_e| \geq \theta_e\}$. Thus this is a special case of our problem where f_e is given by (1).

Applications in Machine Learning. Not only in online advertisement, our influence model over a bipartite graph is of significance in other machine learning applications such as sensor placement (Krause et al., 2008), feature selection (Krause & Guestrin, 2005; Liu et al., 2013), and text summarization (Lin & Bilmes, 2011).

For example, in the standard sensor placement scenario, we aim to allocate sensors so that the number of targets monitored is maximized. To make the system reliable and robust, it is natural to maximize the number of targets monitored by a specified number of sensors, which can be formulated as our threshold model (Example 2). There is another situation in which each sensor may fail to observe a target, and we aim to maximize the number of targets whose probability of being monitored exceeds a specified threshold. This problem can also be formulated as our model (see Examples 3 and 4). In the context of feature selection, the objective is to find a small set that summarizes the feature of a given data as well as possible. For example, in the concept-based text summarization (Filatova & Hatzivassiloglou, 2004), we aim to find a subset X of sentences that maximizes the total credit of concepts covered by X, which is in fact the weighted maximum coverage problem (Example 1). Using our framework, we can further consider finding a subset X of sentences that maximizes the total credit of concepts covered by X a specified number of times.

2.2. Cost-Effective General-Thresholds Coverage

It is observed that, the larger a budget B is, the larger influence we can make to targets. It is then natural to ask how much the budget B is the most effective. That is, we aim to maximize

$$d(X) = \frac{f_G(X)}{c(X)}$$

over all $X \subseteq S$. We call $d(X)$ the *cost-effectiveness* of X. As seen below, the problem has been studied in the context of clustering problems (Examples 5 and 6).

Example 7 (Densest subgraph problem)**.** Maximizing the cost-effectiveness for the densest k-subgraph problem (Example 5) is known as the *densest subgraph problem*. Given a graph (V, F), we are asked to find a subset $X \subseteq V$ that maximizes the *density* $|F(X)|/|X|$, where $F(X)$ is the set of edges in the subgraph induced by X. In contrast to the densest k-subgraph problem, this problem can be solved in polynomial time using a flow-based algorithm (Goldberg, 1984) or an LP-based algorithm (Charikar, 2000). Moreover, Charikar (2000) proved that the greedy algorithm proposed by Asahiro et al. (2000) finds a 2-approximate solution in linear time.

Example 8 (Densest subhypergraph problem)**.** The *densest subhypergraph problem* is to maximize the cost-effectiveness for the densest k-subhypergraph (Example 6). That is, given a hypergraph $H = (V, \mathcal{E})$, we are asked to find a subset $X \subseteq V$ that maximizes $|\mathcal{E}(X)|/|X|$. Huang and Kahng (1995) proposed a flow-based algorithm that solves the problem in polynomial time. Recently, Tsourakakis (2014) extensively studied the l-uniform case (i.e., $|e| = l$ for every $e \in \mathcal{E}$), and proposed efficient exact and approximation algorithms for the problem.

In the above examples, the cost-effectiveness maximization can be solved efficiently. As another example, if f_G is submodular, then it is easy to see that $\arg\max_{s \in S} d(\{s\})$ is an optimal solution. However, it turns out to be NP-hard for a general case. In fact, an instance of the maximum thresholds coverage problem (Example 2) can be reduced to one of the cost-effective general-thresholds coverage problem with polynomial size.

Theorem 1. *The cost-effective general-thresholds coverage problem is NP-hard.*

Finally, let us emphasize that, in a wide variety of machine learning applications described in Section 2.1, the cost-effective maximization is also reasonable, as it is useful to measure how much budget is the most effective.

3. Budget-Constrained Maximization

3.1. Greedy Algorithms

We present two greedy algorithms: the incremental greedy algorithm and the decremental greedy algorithm for the multi-set version of the maximum general-thresholds coverage problem.

Algorithm 1 IncrementalGreedy

Initialize $X_0 \leftarrow \emptyset$ and $i \leftarrow 0$
while $c(X_i) < B$ and $S \neq \emptyset$ **do**
  Find $v \in S \setminus X_i$ that maximizes $\nabla(X_i, v)$
  if $c(X_i \cup \{v\}) \leq B$ **then**
    $X_{i+1} \leftarrow X_i \cup \{v\}$ and $i \leftarrow i + 1$
  else $S \leftarrow S \setminus \{v\}$
return X_i

Algorithm 2 DecrementalGreedy

Initialize $X_{|S|} \leftarrow S$ and $i \leftarrow |S|$
while $c(X_i) > B$ **do**
  Find $v \in X_i$ with smallest contribution $\Delta(X_i, v)$
  $X_{i-1} \leftarrow X_i \setminus \{v\}$ and $i \leftarrow i - 1$
return X_i

We first show that we can reduce a multi-set instance to an instance with unit capacity.

Let $f\colon \mathbb{Z}_+^S \to \mathbb{R}$ be a function over integer lattice. Let $\tilde{S}$ denote $\{(s,i) \mid s \in S, i \in \mathbb{Z}_+\}$. For $X \subseteq \tilde{S}$, we define the *corresponding vector* $x \in \mathbb{Z}_+^S$ as the vector such that $x(s) = \max\{i \in \mathbb{Z}_+ \mid (s,i) \in X\}$, where we note that $x(s) = 0$ if $(s,i) \notin X$ for any i. We define a set function $\tilde{f}\colon 2^{\tilde{S}} \to \mathbb{R}$ by $\tilde{f}(X) = f(x)$ for each $X \subseteq \tilde{S}$. We have the following lemma.

Lemma 1. *An instance of the multi-set version of the maximum general-thresholds coverage problem can be reduced to the one with $u_s = 1$ for each $s \in S$ with pseudo-polynomial size.*

Thus we henceforth present algorithms only for the case where $u_s = 1$ for every $s \in S$.

For $X \subseteq S$, we denote $T(X) = \{t \in T \mid f_t(X_t) \geq \theta_t\}$. The incremental greedy algorithm is described in Algorithm 1. For $s \in S \setminus X$, we define

$$\nabla(X, s) = \frac{\sum_{t \in T(X \cup \{s\}) \setminus T(X)} w_t}{c_s}.$$

It is easy to see that Algorithm 1 runs in $O(\beta|S||E| + |T|)$ time, where β denotes the time to compute the value of f_t.

On the other hand, the decremental greedy algorithm is presented in Algorithm 2. For $s \in X$, we define the *contribution of s in X* by

$$\Delta(X, s) = \frac{\sum_{t \in T(X) \cap \Gamma(s)} w_t}{c_s}.$$

Clearly, Algorithm 2 runs in $O(|S|^2 + |T| + \beta|E|)$ time. The following theorem states that the algorithm can be implemented to run in almost linear time.

Theorem 2. *Algorithm 2 runs in $O(|S| + \beta|E| + W)$ time when $c_s = 1$ for each $s \in S$ and w_t is an integer for each $t \in T$, where $W = \sum_{t \in T} w_t$. For the general case, Algorithm 2 runs in $O(|S| \log |S| + |T| + \beta|E|)$ time.*

3.2. Submodular Thresholds Coverage

In this section, a threshold function f_t is limited to a $[0,1]$-valued monotone submodular function over integer lattice. An example of a monotone submodular function can be found in Examples 1, 2, and 4.

Moreover, for each target $t \in T$, a threshold θ_t is supposed to be chosen uniformly at random from the interval $[0,1]$. The objective is to find an integer vector $x = (x_s)_{s \in S}$ that maximizes the *expected number* of targets $t \in T$ satisfying $f_t(x_t) \geq \theta_t$, subject to a total budget constraint. Specifically, the objective function is represented as $\sum_{t \in T} f_t(x_t)$. Since the sum of monotone submodular functions is monotone and submodular, the problem is maximizing a monotone submodular function over integer lattice subject to a total budget constraint. Therefore, it follows from Soma et al. (2014) that we can find an $(e/(e-1))$-approximate solution in pseudo-polynomial time.

We here present a simpler different approach from Soma et al. (2014). The main idea is to use the reduction in Lemma 1. The following theorem guarantees that the reduced instance preserves submodularity.

Theorem 3. *Let $f\colon \mathbb{Z}_+^S \to \mathbb{R}$ be a function over integer lattice. Then f is monotone and submodular if and only if the set function $\tilde{f}$ is monotone and submodular.*

Let $\tilde{S} = \{(s,i) \mid s \in S, i \in \mathbb{Z}_+\}$, and let $\tilde{f}$ be a monotone submodular set function defined in Theorem 3. We define $c(s,i) = c_s i$ for $(s,i) \in \tilde{S}$, and consider the problem of maximizing $\tilde{f}(X)$ subject to $\sum_{(s,i) \in X} c(s,i) \leq B$. Then an α-approximate solution for this problem yields an α-approximate one for the original problem. Since the well-known greedy algorithm (with partial enumeration) gives an $(e/(e-1))$-approximate solution for the reduced problem (Sviridenko, 2004), we have an $(e/(e-1))$-approximation algorithm for the original problem.

Remark 1. Theorem 3 provides a general framework for problems related to a monotone submodular function over integer lattice to reduce to the counterpart on a set function. In particular, this gives a simpler proof for the approximation result on maximizing a monotone submodular function over integer lattice (Soma et al., 2014). Consider maximizing a monotone submodular function $f(x)$ over integer lattice subject to a budget constraint $\sum_{i \in S} c_i x_i \leq B$ and a capacity $x \leq u$. Then we can reduce to the one of maximizing a monotone submodular set function subject to a budget constraint, which yields an $(e/(e-1))$-approximation algorithm in pseudo-polynomial time (see the Supplemen-

tary Material for details). Note that our reduction implies a pseudo-polynomial-time algorithm slightly different from Soma et al. (2014) in the sense that our algorithm uses different marginal return to choose the next one.

The same reduction is applicable to the submodular set cover problem (Wolsey, 1984). Using our reduction framework, we obtain a logarithmic-approximation algorithm, running in pseudo-polynomial time, for the multi-set version of the submodular set cover problem. Furthermore, we can apply the same reduction for more general problems such as multi-set generalizations of the submodular cost submodular cover and submodular cost submodular knapsack problems (Iyer & Bilmes, 2013).

4. Cost-Effectiveness Maximization

4.1. Fast Greedy Approximation

In this section, we prove that Algorithm 2 essentially returns an approximate solution for the cost-effective general-thresholds coverage problem. Define γ to be the maximum degree of a node in T, i.e., $\gamma = \max_{t \in T} |\Gamma(t)|$.

Our algorithm repeatedly removes a source node with the minimum contribution in the remaining graph, and then returns the subset with maximum cost-effectiveness during the process. That is, we execute Algorithm 2 with $B = 0$ to obtain a sequence of subsets $X_{|S|}, \dots, X_0$, and return $\arg\max_{i=0,\dots,|S|} d(X_i)$. This runs in $O(|S| \log |S| + |T| + \beta|E|)$ time by Theorem 2. We have the following theorem.

Theorem 4. *The above algorithm is a γ-approximation algorithm for the cost-effective general-thresholds coverage problem.*

The key observation is that, in the sequence $X_{|S|}, \dots, X_0$, if X_i is the minimum subset containing an optimal solution for the problem, then the contribution $\Delta(X_i, s)$ is at least the optimal cost-effectiveness for any $s \in X_i$. This implies that X_i is γ-approximation.

4.2. Linear-Programming-based Approximation

In this section, we propose an LP-based approximation algorithm for the case where $c_s = 1$ for each source $s \in S$ and $f_t(X) = |X|$ for each target $t \in T$. Define $p_t = |\Gamma(t)| - \theta_t + 1$ for each $t \in T$ and $p = \max_{t \in T} p_t$.

We introduce a variable x_s for each $s \in S$ and a variable y_t for each $t \in T$. Consider the following linear programming problem (LP):

$$\text{max.} \sum_{t \in T} w_t y_t$$

$$\text{s. t.} \sum_{s \in J} x_s \geq y_t \ (\forall t \in T, \forall J \subseteq \Gamma(t) \text{ with } |J| = p_t);$$

$$\sum_{s \in S} x_s \leq 1; \quad x_s \geq 0 \ (\forall s \in S); \quad y_t \geq 0 \ (\forall t \in T).$$

Note that (LP) has an exponential number of constraints. However, we can check the feasibility in polynomial time, because the first constraints are satisfied if and only if the sum of the smallest p_t variables corresponding to the sources in $\Gamma(t)$ is at least y_t for each $t \in T$. Therefore, (LP) can be solved in polynomial time using the ellipsoid method (Grötschel et al., 1981).

We first observe that we can construct a feasible solution of (LP) with objective value at least $d(X)$ from a subset $X \subseteq S$.

Lemma 2. *For any $X \subseteq S$, there exists a feasible solution of* (LP) *whose objective value is greater than or equal to $d(X)$.*

Conversely, let $(\overline{x}, \overline{y})$ be a feasible solution of (LP). For a real parameter $r \geq 0$, we define a sequence of subsets $X(r) = \{s \in S \mid \overline{x}_s \geq r/p\}$. Then, taking the integral of $d(X(r))$ with respect to r from 0 to 1, we see that $X(r)$ for some $r \in [0, 1]$ has large cost-effectiveness.

Lemma 3. *For any feasible solution $(\overline{x}, \overline{y})$ of* (LP) *with objective value λ, there exists $s \in S$ that satisfies $d(X(p\overline{x}_s)) \geq \lambda/p$.*

Our LP-based algorithm is designed as follows. It solves (LP) to obtain an optimal solution $(\overline{x}, \overline{y})$, and then returns $X(p\overline{x}_s)$ that maximizes the cost-effectiveness over $s \in S$. From the above lemmas, we have the following theorem.

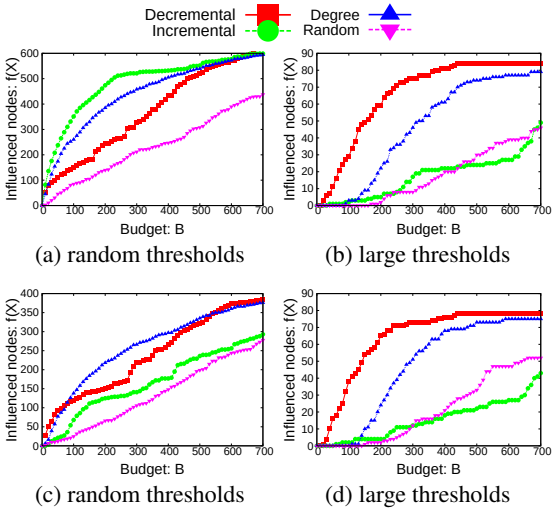
Theorem 5. *The LP-based algorithm is a polynomial-time p-approximation algorithm for the cost-effective thresholds coverage problem.*

5. Experiments

To evaluate the performance of our algorithms, we have conducted simulation on a real-world network and large-scale networks. All experiments were performed on a Windows PC with Intel Core i7 2.40 GHz CPU and 16 GB RAM. Our algorithms were implemented in C++.

We compare algorithms Incremental (Algorithm 1) and Decremental (Algorithm 2) with two baseline algorithms Degree and Random. Degree selects nodes in decreasing degree order, and Random selects nodes randomly. Remark that the LP-based algorithm in Section 4.2 was not implemented because the ellipsoid method for (LP) is required, which is expected to be impractical.

We use as f_t the functions (2) of the target-side model and (3) of the source-side model. A sequence of probabilities in the functions is set randomly from the interval $[0, 0.1]$. Note that the function (3) is submodular, while (2) is not. We define $c_s = 1$ for each source s and $w_t = 1$ for each target t. We set $u_s = 1$ as we can reduce any instance into the case.



(a) random thresholds (b) large thresholds

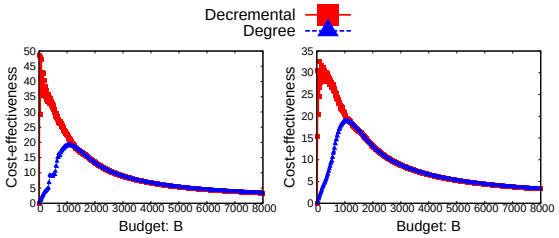
(c) random thresholds (d) large thresholds

Figure 1. Results for Yahoo! Bidding Data. (a) and (b) are on the source-side model, and (c) and (d) are on the target-side model.

As a real-world network, we use the Yahoo! Search Marketing Advertiser Bidding Data (Yahoo!) as in (Soma et al., 2014). It is a bipartite graph between 1,000 search keywords and 10,475 accounts, where each edge represents one "bid" to advertisement on the keyword. The graph has 52,567 edges (representing 'who bid at least once to what' relation). We set threshold θ_t for each target t in the following two ways: random thresholds means θ_t is chosen randomly from the interval $[0, 1]$, and large thresholds means θ_t is chosen randomly from the interval $[0.5, 1]$.

Figure 1 shows influence spreads with respect to a budget size for each algorithm. The results depend on the settings. For the source-side model with random thresholds, Incremental always returns the best values. This result matches the theoretical guarantee for the submodular case, as the expected value of the objective value is submodular. When thresholds are large, Decremental always returns the best values. The other algorithms are of poor quality particularly when B is small. For the target-side model, Decremental returns the best values in most cases, except for the random thresholds case when B is of medium size. For all settings, Decremental took within one second, similarly to Random and Degree, while Incremental took much time (a few hours). In summary, it is observed that Decremental is fast and of the best quality when thresholds are large. When the objective function is close to submodular, Incremental is better but slower.

We also measured the maximum cost-effectiveness obtained by Decremental for each setting. Recall that when f_G is submodular, there is an optimal solution of size one. For the source-side model with random thresholds, the output of size one achieves the maximum cost-effectiveness, which implies that this setting has a nearly-submodular objective function. For the target-side model with random thresholds and the source-side and target-side models with large thresholds, the outputs with maximum cost-effectiveness have size 8, 130, and 95, respectively.



(a) source-side model (b) target-side model

Figure 2. Cost-effectiveness for large-scale networks

To evaluate for larger-scale networks, we have also run our experiments on artificially generated graphs. We set $|S| =$ 20,000 and $|T| =$ 200,000 with around 2,000,000 edges between them. The graphs are generated so that the degree distribution of the source nodes obeys the *power law* with exponent $\gamma = 2.0$. After assigning the degree $\deg(s)$ to each source node s, it is connected to $\deg(s)$ nodes chosen uniformly from T. We set $\theta_t = 0.8$ for every target t. We executed only Decremental and Degree as they outperformed the other algorithms in the Yahoo! Search Marketing Advertiser Bidding Data with large thresholds.

Figure 2 demonstrates the cost-effectiveness of the solutions obtained by each algorithm. Decremental returns a better solution than Degree, whose size with maximum cost-effectiveness is 28 for the source-side model and 37 for the target-side model. Also, Decremental returns better influence spreads than Degree. Decremental took only a few minutes for both models.

6. Conclusion

In this paper, we have introduced a new influence model over a bipartite graph. The model captures the different sensitivities of customers to advertisements as thresholds. We have first discussed the maximum general-thresholds coverage problem, and proposed two greedy algorithms; the incremental greedy algorithm extends the one for the submodular case, while the decremental greedy algorithm runs in almost linear time. Numerical experiments suggest to use the incremental one only for the nearly-submodular case, and the decremental one for the other cases. We have then introduced another type of optimization problem, the cost-effective general-thresholds coverage problem, to analyze the performance of our campaign. For the problem, we have developed an almost linear-time approximation algorithm, and furthermore, an LP-based better-approximation algorithm for a special case.

Acknowledgments

The authors thank the reviewers for useful comments and suggestions to improve the paper. Also, the authors thank Yusuke Kobayashi for fruitful discussion on this topic. AM is supported by a Grant-in-Aid for JSPS Fellows (No. 26-11908), TF is supported by a Grant-in-Aid for Young Scientists (B) (No. 25730008), and NK is supported by a Grant-in-Aid for Young Scientists (B) (No. 25730001). This work was supported by JST, ERATO, Kawarabayashi Large Graph Project.

References

Alon, N., Gamzu, I., and Tennenholtz, M. Optimizing budget allocation among channels and influencers. In *WWW '12: Proceedings of the 21st International Conference on World Wide Web*, pp. 381–388, 2012.

Asahiro, Y., Iwama, K., Tamaki, H., and Tokuyama, T. Greedily finding a dense subgraph. *Journal of Algorithms*, 34(2):203–221, 2000.

Bhaskara, A., Charikar, M., Chlamtac, E., Feige, U., and Vijayaraghavan, A. Detecting high log-densities: An $O(n^{1/4})$ approximation for densest k-subgraph. In *STOC '10: Proceedings of the 42nd ACM Symposium on Theory of Computing*, pp. 201–210, 2010.

Charikar, M. Greedy approximation algorithms for finding dense components in a graph. In *APPROX '00: Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization*, pp. 84–95, 2000.

Domingos, P. and Richardson, M. Mining the network value of customers. In *KDD '01: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 57–66, 2001.

Feige, U., Peleg, D., and Kortsarz, G. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

Filatova, E. and Hatzivassiloglou, V. A formal model for information selection in multi-sentence text extraction. In *COLING '04: Proceedings of the 20th International Conference on Computational Linguistics*, pp. 397–403, 2004.

Goldberg, A. V. Finding a maximum density subgraph. Technical report, University of California Berkeley, 1984.

Goldenberg, J., Libai, B., and Muller, E. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12(3):211–223, 2001a.

Goldenberg, J., Libai, B., and Muller, E. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001b.

Granovetter, M. Threshold models of collective behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978.

Grötschel, M., Lovász, L., and Schrijver, A. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

Hochbaum, D. S. Approximating covering and packing problems: Set cover, vertex cover, independent set, and related problems. In Hochbaum, D. S. (ed.), *Approximation Algorithms for NP-hard Problems*, pp. 94–143. PWS Publishing Co., 1997.

Huang, D. J. H. and Kahng, A. B. When clusters meet partitions: New density-based methods for circuit decomposition. In *EDTC '95: Proceedings of the 1995 European Conference on Design and Test*, pp. 60–64, 1995.

Iyer, R. K. and Bilmes, J. A. Submodular optimization with submodular cover and submodular knapsack constraints. In *NIPS '13: Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, pp. 2436–2444, 2013.

Kempe, D., Kleinberg, J., and Tardos, É. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 137–146, 2003.

Kempe, D., Kleinberg, J., and Tardos, É. Influential nodes in a diffusion model for social networks. In *ICALP '05: Proceedings of the 32nd International Colloquium on Automata, Languages and Programming*, pp. 1127–1138, 2005.

Krause, A. and Golovin, D. Submodular function maximization. In Bordeaux, L., Hamadi, Y., and Kohli, P. (eds.), *Tractability: Practical Approaches to Hard Problems*, pp. 71–104. Cambridge University Press, 2014.

Krause, A. and Guestrin, C. Near-optimal nonmyopic value of information in graphical models. In *UAI '05: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pp. 324–331, 2005.

Krause, A., Singh, A., and Guestrin, C. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

Lin, H. and Bilmes, J. A. A class of submodular functions for document summarization. In *HLT '11: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 510–520, 2011.

Liu, Y., Wei, K., Kirchhoff, K., Song, Y., and Bilmes, J. A. Submodular feature selection for high-dimensional acoustic score spaces. In *ICASSP '13: Proceedings of the 38th International Conference on Acoustics, Speech and Signal Processing*, pp. 7184–7188, 2013.

Lu, Z., Zhang, W., Wu, W., Fu, B., and Du, D. Approximation and inapproximation for the influence maximization problem in social networks under deterministic linear threshold model. In *ICDCS '11: Proceedings of the 31st International Conference on Distributed Computing Systems*, pp. 160–165, 2011.

Lu, Z., Zhang, W., Wu, W., Kim, J., and Fu, B. The complexity of influence maximization problem in the deterministic linear threshold model. *Journal of Combinatorial Optimization*, 24(3):374–378, 2012.

Minoux, M. Accelerated greedy algorithms for maximizing submodular set functions. In *Proceedings of the 8th IFIP Conference on Optimization Techniques*, pp. 234–243, 1978.

Mossel, E. and Roch, S. Submodularity of influence in social networks: From local to global. *SIAM Journal on Computing*, 39(6):2176–2188, 2010.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

Papailiopoulos, D. S., Mitliagkas, I., Dimakis, A. G., and Caramanis, C. Finding dense subgraphs via low-rank bilinear optimization. In *ICML '14: Proceedings of the 31st International Conference on Machine Learning*, pp. 1890–1898, 2014.

Richardson, M. and Domingos, P. Mining knowledge-sharing sites for viral marketing. In *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 61–70, 2002.

Singh, A. P., Guillory, A., and Bilmes, J. A. On bisubmodular maximization. In *AISTATS '12: Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pp. 1055–1063, 2012.

Soma, T., Kakimura, N., Inaba, K., and Kawarabayashi, K. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *ICML '14: Proceedings of the 31st International Conference on Machine Learning*, pp. 351–359, 2014.

Sviridenko, M. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41 – 43, 2004.

Tsourakakis, C. E. A novel approach to finding near-cliques: The triangle-densest subgraph problem. *arXiv:1405.1477*, 2014.

Wolsey, L. A. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4): 385–393, 1984.

Yahoo! Webscope dataset: A1 - Yahoo! Search Marketing Advertiser Bidding Data, version 1.0. http://webscope.sandbox.yahoo.com/catalog.php?datatype=a.