Robust Budget Allocation

by

Ashley Mimi Hou

A thesis submitted in partial fulfillment of the

requirements for the degree of

Master of Science

(Electrical and Computer Engineering)

at the

UNIVERSITY OF WISCONSIN - MADISON

2019

Committee in charge:

Line A. Roald, Assistant Professor, Electrical and Computer Engineering, Chair
Po-Ling Loh, Associate Professor, Statistics, Co-chair
Varun Jog, Assistant Professor, Electrical and Computer Engineering

Robust Budget Allocation

Abstract

Robust Budget Allocation

by

Ashley Mimi Hou

We consider at the problem of optimal budget allocation, where we are given a fixed budget to distribute amongst sources in order to maximally influence targeted individuals. In practice, situations may arise where the influence process is unknown to us, motivating us to look at the robust budget allocation problem. In this setting, influence functions, which capture the expected number of influenced targets, are known only up to an uncertainty set and the goal is to find an allocation that maximizes the worst-case influence function.

We extend the results of previous work on the related problem of robust submodular maximization to the integer lattice domain, formulating a polynomial-time algorithm approximation algorithm, SaturateDR. We prove hardness of approximation results for both the robust budget allocation problem as well as the bi-criteria approximation obtained by SaturateDR. Finally, we present experimental results that demonstrate our robust solution performs favorably in comparison to non-robust methods, with respect to influence obtained as well as runtime.

Acknowledgments

First and foremost, I would like to express my gratitude towards my advisor Po-Ling Loh. Her guidance, expertise, and patience have been invaluable to me as I began my graduate career. I would like to thank Varun Jog for serving on my committee as well as everyone in the Jog-Loh lab for their insightful research discussions. I am thankful to Line Roald, with whom I greatly look forward to developing many new and exciting research collaborations.

I would like to extend my thanks to all my friends, both in Madison and beyond, for their continued friendship and support. Most of all, I am thankful to my family, who have been the foundation of my education.

Contents

Chapter 1

Introduction

For many companies, marketing and advertising are essential tools for influencing customers. The recent emergence of new media platforms, such as social networks and online search advertising, has presented an explosion of possible marketing channels. A key decision faced by companies is how to optimally distribute their limited budget among various platforms in order to maximally influence potential customers. To capture this issue, Alon et al. [1] introduced the budget allocation problem, which models this setting using a bipartite graph that consists of source nodes (representing marketing channels), target nodes (representing customers), and edges (representing interactions between channels and individuals). The likelihood of influencing an individual depends on not only the channel's ability to reach an individual, but also the amount of budget allocated to the channel.

Closely related to the budget allocation problem is the influence maximization problem, which studies influence or information diffusion over social networks. First introduced by Richardson and Domingos, 2001 [2] 2002 [3], influence maximization aims to select a (small) set of individuals in a network to directly influence such that after information diffusion occurs across the network, the resulting number of influenced individuals is maximized. Budget allocation can be viewed as influence maximization over a bipartite graph. However, a key difference that distinguishes the two problems is that in budget allocation, multiple units of budget can be assigned to source nodes. This allows more the budget allocation model more flexibility in capturing realistic scenarios where there may exist varying levels of influence rather than a solely binary choice for each node.

Both budget allocation and influence maximization are NP-hard to solve. For the case of influence maximization, Kempe et al., 2003 [4] demonstrated that a greedy algorithm can achieve a $(1 - 1/e)$-approximation by leveraging the submodularity property of the influence function, which captures the expected number of influenced nodes at the end of the diffusion process. Due to the inclusion of budgets, influence functions for budget allocation cannot be capture using set functions and must be defined over the integer lattice. Using the integer lattice analogue of submodularity, diminishing returns (DR)-submodularity, Soma et al., 2014 [5] also achieve a $(1 - 1/e)$-approximation using a greedy algorithm.

For standard formulations of budget allocation and influence maximization, both the

diffusion process and probabilities of influence transmission between nodes are assumed to be known. However, this is an unrealistic assumption. For example, it is difficult to know the exact likelihood that an advertisement will convince an individual to purchase a product. Previous works have looked to infer model parameters that describe transmission probabilities using historical observations (Saito et al., 2008 [6], Netrapalli and Sanghavi, 2012 [7], Gomez-Rodriguez et al., 2012 [8], Gomez-Rodriguez et al., 2014 [8]). A drawback to such methods is that they do not address the issue of model misspecification. In fact, it has been demonstrated that information diffusion observed from real-life data does not typically occur according to the models commonly used in theory (Hu et al., 2018 [9]). Motivated by these issues, we look at budget allocation through the perspective of robust optimization, where the influence function is only known up to an uncertainty set, and we aim to maximize the worst-case influence function. This problem is know as the *robust budget allocation* problem.

The main contributions of this thesis are

(i) Formulating a polynomial-time, bi-criteria approximation algorithm, SaturateDR, for robust budget allocation by extending previous work in robust submodular maximization to the integer lattice domain.

(ii) Providing hardness of approximation results for the robust budget allocation problem. and the bi-criteria guarantee achieved by SaturateDR.

(iii) Experimentally demonstrating the benefits of our robust solution in comparison to non-robust methods.

The remainder of the thesis is organized as follows: Chapter 2 introduces technical background and foundational work on budget allocation, influence maximization, and submodular optimization referenced throughout the thesis. We formally define the robust budget allocation problem and present related works in Chapter 3. We present our algorithm and theoretical results in Chapter 4. In Chapter 5, we present and discuss experimental results. Finally, we conclude with suggestions for future work in Chapter 6.

Notation: We use boldface letters (i.e., $\mathbf{x}$) to represent vector-valued variables, where $\mathbf{x}(s)$ represents the s-th element of $\mathbf{x}$. Given two vectors $\mathbf{x}, \mathbf{y}$, $\mathbf{x} \leq \mathbf{y}$ denotes $\mathbf{x}(i) \leq \mathbf{y}(i)$ for all i. We use $\vee$ and $\wedge$ to represent the coordinate-wise maximum and minimum, respectively. Let $f(\mathbf{y}|\mathbf{x}) := f(\mathbf{x}+\mathbf{y}) - f(\mathbf{x})$ denote the marginal gain in f of adding $\mathbf{y}$ to $\mathbf{x}$.

Chapter 2

Background

This chapter introduces technical background, concepts, and notation used throughout this thesis. Although our results concern the budget allocation problem, since most previous work in budget allocation is derived from the related problem of influence maximization, we discuss this setting, as well.

2.1 Budget allocation problem

We consider the formulation of the budget allocation problem introduced by Alon et al., 2012 [1]. The setting consists of a directed bipartite graph, $G = (S, T; E)$, where S is a set of *source nodes*, T is a set of *target nodes*, and $E \subseteq S \times T$ is the *edge set*. Each source node $s \in S$ has a *nodal budget capacity*, $\mathbf{r}(s) \in \mathbb{Z}_+$, and there exists a global *budget*, $B \in \mathbb{Z}_+$. By allotting some amount of budget to a source node, it can *influence* or *activate* target nodes connected to it via an edge. The aim of the budget allocation problem is to distribute the budget amongst source nodes such that the nodal budget capacities are satisfied and the expected number of influenced or active target nodes is maximized.

Formally, let $\mathbf{x} \in \mathbb{Z}_+^{|S|}$ represent a *budget allocation*, where $\mathbf{x}(s)$ indicates the amount of budget allocated to source node $s \in S$. Let $f : \mathbb{Z}_+^{|S|} \to \mathbb{R}_+$ be the *influence function*, which indicates the expected number of influenced target nodes achieved by a budget allocation. The exact mathematical definition of f depends on which influence diffusion model is employed, and will be discussed in the subsequent section. Let $c : \mathbb{Z}_+^{|S|} \to \mathbb{R}$ be the *cost function*, which denotes the cost of selecting a budget allocation. We consider additive cost functions of the form $c(\mathbf{x}) = \sum_{s \in S} \mathbf{x}(s)$, known as *knapsack constraints*. The budget allocation problem can be expressed as

$$\max_{\mathbf{x} \in \mathbb{Z}_+^{|S|}} f(\mathbf{x}) \quad \text{s.t.} \quad c(\mathbf{x}) \leq B \qquad (2.1)$$
$$\mathbf{x} \leq \mathbf{r}.$$

If the nodal budget capacity of every source node is one, i.e., $\mathbf{r}(s) = 1$ for all $s \in S$, budget allocation reduces to a case of influence maximization. Influence maximization is defined on a (general) directed graph, $G = (V; E)$, where V is a set of nodes and E is the edge set. Similarly, nodes can influence or activate one another via edges. The goal of influence maximization is to select a small *seed set* of nodes to initially activate such that their expected influence on the graph is maximized.

Let A be the selected seed set, parameter $k \in \mathbb{Z}_+$ be a cardinality constraint on A, and $f : 2^V \to \mathbb{R}$ be the influence function. The influence maximization problem can be written as

$$\max_{A \subseteq V} f(A) \quad \text{s.t.} \quad |A| \leq k. \tag{2.2}$$

Although there exist instances of budget allocation that are equivalent to instances of influence maximization, we highlight that neither problem is a special case of the other, the main differences being:

(i) Multiple units of budget can be assigned to sources node in budget allocation. As a result, budget allocation must be studied on the integer lattice, whereas influence maximization can be captured using set functions.

(ii) In budget allocation, we restrict consideration to bipartite graphs, where only source nodes can be allotted budget and only target nodes can be influenced or activated. Given this graph structure, unlike in influence maximization, the propagation of influence among nodes is not examined.

2.2 Influence diffusion models

Influence diffusion models capture the process by which nodes are influenced or activated. In this thesis, we restrict ourselves to *progressive*, *discrete-time* models. In progressive models, inactive nodes can be activated, but active nodes cannot be deactivated. In discrete-time models, time unfolds in discrete time steps, $\tau = 1, \ldots, T$. If a node is activated in time step τ, it will appear as active during time step $\tau + 1$. In the following discussion, we outline the two models that are primarily used in this thesis: the independent cascade (IC) model and the triggering model.

2.2.1 Independent cascade model

We consider the formulation by Kempe et al., 2003 [4] for influence maximization, where each edge, $(u, v) \in E$, has an associated parameter $p_{u,v}$ that denotes the probability that node u successfully activates node v. Let $\Gamma_{\text{out}}(v)$ denote the set of outgoing neighbors of v (i.e., all nodes u where there exists an edge from v to u) and $\Gamma_{\text{in}}(v)$ denote the set of incoming neighbors of node v (i.e., all nodes u where there exists an edge from u to v). After an initial seed set is selected, diffusion occurs as follows: if node u becomes active at time τ, it has

one independent attempt to activate each of its inactive outgoing neighbors, $v \in \Gamma_{\text{out}}(u)$, and succeeds with probability $p_{u,v}$. This process continues until no further activations are possible.

Based on this model, Alon et al., 2012 [1] introduced the source-side and target-side influence models to capture the budget allocation setting. We consider a generalized version used by Soma et al., 2014 [5]. Here, the amount of budget allotted to a source node controls the number of independent attempts it has to activate neighboring target nodes. The probability that a target node t becomes active is

$$f_t(\mathbf{x}) = 1 - \prod_{s \in \Gamma_{\text{in}}(t)} (1 - p_{s,t})^{\mathbf{x}(s)}, \tag{2.3}$$

where $p_{s,t}$ is the probability that node s influences node t. We obtain the influence function by taking the expectation over all target nodes:

$$f(\mathbf{x}) = \sum_{t \in T} f_t(\mathbf{x}). \tag{2.4}$$

Alon et al. also consider a generalization to the case where the underlying graph is not restricted to be bipartite. In this setting, budget can be allotted to any node and each node has a self-loop. As a result, each node is considered to be a neighbor of itself and can influence itself as a result of the budget allotted to it.

2.2.2 Triggering model

In both problems, the IC model can be generalized by the triggering model (Kempe et al., 2003 [4]). For each instance of the diffusion process, each node v independently chooses a triggering set, $R(v)$, according to a probability distribution over its incoming neighbors. Once diffusion is initialized with a seed set, an inactive node becomes active if its triggering set contains an active node. Triggering sets can alternatively be interpreted as capturing which edges are "live": if $u \in R(v)$, then there exists a "live" edge by which u can influence v.

For influence maximization, we obtain the IC model by choosing the probability $u \in R(v)$ to be $p_{u,v}$. For budget allocation, we consider source nodes where $\mathbf{x} > 0$ to be the initially activated seed set. Let the triggering sets of all source nodes be empty. The probability that source node, s, influences target node, t, is $1 - (1 - p_{s,t})^{\mathbf{x}(s)}$ in the IC model, so we let this be the probability that $s \in R(t)$.

To express the triggering model mathematically, let $G \in \mathcal{G}$ be one possible fixed selection of triggering sets for all nodes, $\Pr(G)$ be the probability of G occurring, and $f_G(\mathbf{x})$ be the expected number of influenced target nodes achieved $\mathbf{x}$ under G. The influence function is

$$f(\mathbf{x}) = \sum_{G \in \mathcal{G}} \Pr(G) \cdot f_G(\mathbf{x}). \tag{2.5}$$

An alternative generalization of the triggering model, called the budgeted triggering model, was introduced by Avigdor-Elgrabli et al., 2015 [10]. This model considers general graphs,

allowing it to capture both the notions of budgets and propagation of influence amongst nodes. However, it is not equivalent to the discussed IC model for budget allocation.

2.3 Influence function properties

It is known to be NP-hard to find optimal solutions for budget allocation and influence maximization under most studied diffusion models, including the IC and triggering models discussed (Alon et al., 2012 [1], Kempe et al., 2003 [4]). Fortunately, both models admit influence functions with properties that allow for efficient approximations with strong theoretical guarantees.

The first property is *monotonicity*, which imposes the condition that increasing the function's input cannot decrease the function's output.

Definition 2.3.1. (**Monotonicity**) An integer lattice function $f : \mathbb{Z}_+^{|S|} \to \mathbb{R}_+$ is *monotone increasing* if $f(\mathbf{x}) \leq f(\mathbf{y})$ whenever $\mathbf{x} \leq \mathbf{y}$.

The second property is *submodularity*, a structural property that formalizes the notion of diminishing returns: the marginal benefit of adding an additional item to a smaller set must be at least as high as the marginal benefit of adding the same item to a larger set. Since the diminishing returns property naturally arises in many real-world scenarios, submodular functions are widely applicable to a variety of domains and are a well-studied area of interest. Submodularity is typically associated with set functions, where it is defined as:

Definition 2.3.2. (**Submodular set function**) A set function $f : 2^{|V|} \to \mathbb{R}_+$ is *submodular* if and only if either of the following equivalent conditions hold:

(i) For any two sets $S \subseteq T \subseteq V$ and any $x \notin T$,

$$f(T \cup \{x\}) - f(T) \leq f(S \cup \{x\}) - f(x). \tag{2.6}$$

(ii) For any sets $S, T \subseteq V$,

$$f(S \cup T) + f(S \cap T) \leq f(S) + f(T). \tag{2.7}$$

To be applicable to the budget allocation problem, we need to consider submodular functions over a bounded subset of the integer lattice, $\mathbb{Z}_+^{|S|}$. We define *lattice submodularity*, a direct generalization of Equation 2.7, as follows:

Definition 2.3.3. (**Lattice submodular function**) A function on the integer lattice $f : \mathbb{Z}_+^{|S|} \to \mathbb{R}_+$ is *lattice submodular* if for all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_+^{|S|}$, f satisfies

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \vee \mathbf{y}) + f(\mathbf{x} \wedge \mathbf{y}). \tag{2.8}$$

Clearly, we can reduce lattice submodularity to submodularity over sets by restricting consideration to only vectors with entries that take values in $\{0, 1\}$. Unlike submodularity on set functions, lattice submodularity does *not* imply the diminishing returns property, which is the salient feature of submodularity. We consider the following stronger notion of submodularity:

Definition 2.3.4. (**Diminishing returns submodular (DR-submodular) function**) A function, $f : \mathbb{Z}_+^{|S|} \to \mathbb{R}_+$, is *DR-submodular* if f satisfies

$$f(\mathbf{x} + \chi_s) - f(\mathbf{x}) \geq f(\mathbf{y} + \chi_s) - f(\mathbf{y}) \tag{2.9}$$

for $\mathbf{x} \leq \mathbf{y}$ and $s \in S$, where χ_s is the unit vector with entry 1 in the s-th component.

We highlight a few useful properties of monotone DR-submodular functions used in this thesis:

(i) The class of monotone DR-submodular functions remains closed under non-negative linear combinations.

(ii) Truncations of monotone DR-submodular functions remain monotone DR-submodular, i.e., if $f(\mathbf{x})$ is monotone DR-submodular and $c \geq 0$ is a constant, then $g(\mathbf{x}) = \min\{f(\mathbf{x}), c\}$ is monotone DR-submodular.

Finally, we note that submodularity has been studied on continuous domains as well, where, as in the case of integer lattice functions, submodularity is not equivalent to the diminishing returns property. Here, functions are defined on subsets of $\mathbb{R}^n$, where $n \in \mathbb{Z}_+$, of the form $\mathcal{X} = \prod_{i=1}^n \mathcal{X}_i$, where each $\mathcal{X}_i$ is a compact subset of $\mathbb{R}_+$.

Definition 2.3.5. (**Continuous DR-submodular function**) A continuous function $F : \mathcal{X} \to \mathbb{R}_+$ is *DR-submodular* if and only if

(a) For any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, scalar $k \in \mathbb{R}_+$, basis vector $\mathbf{e} \in \mathbb{R}^n$ such that $\mathbf{x} \leq \mathbf{y}$ and $\mathbf{x}+k\mathbf{e}, \mathbf{y}+k\mathbf{e} \in \mathcal{X}$,

$$F(\mathbf{x} + k\mathbf{e}) - F(\mathbf{x}) \geq F(\mathbf{y} + k\mathbf{e}) - F(\mathbf{y}).$$

If F is twice-differentiable, F is DR-submodular if and only if all entries of its Hessian are non-positive, i.e.,

$$\nabla_{ij}^2 F(\mathbf{x}) \leq 0,$$

for all $\mathbf{x} \in \mathcal{X}$ and $i, j \in [n]$.

2.4 Submodular function maximization

Under the IC model, the influence function in budget allocation (Equation 2.4) is a monotone DR-submodular function. As a result, budget allocation can be viewed as an instance of the more general problem of maximizing a monotone DR-submodular function subject to a knapsack constraint (Soma et al., 2014 [5], Hatano et al., 2015 [11]). Likewise, under the IC model, the influence function for influence maximization is a monotone submodular set function and therefore influence maximization is a case of monotone submodular maximization subject to a cardinality constraint (Kempe et al., 2003 [4]).

Monotone submodular function maximization is generally NP-hard in both the set function and integer lattice domains. However, due to submodularity, there exist polynomial-time approximation algorithms that obtain near-optimal solutions. Since most algorithms for budget allocation are derived from algorithms for influence maximization, we discuss methods for both budget allocation and influence maximization in the broader context of constrained monotone submodular and DR-submodular maximization.

2.4.1 Greedy methods

Submodular set function maximization encompasses problems where the goal is to find a subset of elements, $A \subseteq V$, that maximizes a monotone submodular function, $f : 2^V \to \mathbb{R}_+$, subject to a constraint on A. Even for a simple cardinality constraint, $|A| \leq k$, this problem is NP-hard, but can be approximated to a factor of $(1 - 1/e)$ using a *greedy algorithm* (Nemhauser et al., 1978 [12]). The greedy algorithm (Algorithm 1), incrementally builds a solution by iteratively adding the element with the largest marginal increase in f to the current set. By establishing the submodularity of influence functions for various diffusion models, Kempe et al., 2003 [4] use the greedy algorithm to achieve a $(1 - 1/e)$-approximation for the influence maximization problem.

The greedy algorithm can also be applied to maximization problems with more complex constraints. A 1/2-approximation can be achieved for a general matroid constraint (Nemhauser et al., 1978 [12]). For a knapsack constraint, the addition of an enumeration step allows the greedy algorithm to obtain a $(1 - 1/e)$-approximation (Sviridenko, 2004 [13]).

Algorithm 1 Greedy Algorithm

1: **Initialize**: $A_0 \leftarrow \emptyset$
2: **for** $i = 0, \ldots, k - 1$ **do**
3: $\quad e^* = \arg\max_{e \in V \setminus A_i} f(A_i \cup \{e\}) - f(A_i)$
4: $\quad A_{i+1} = A_i \cup \{e^*\}$
5: **Return**: A_k

For approximating influence maximization, the greedy algorithm suffers from two main drawbacks. The first being that it requires $\mathcal{O}(|V|\, k)$ function evaluations. In practice, this

can be reduced with *lazy evaluations*, which uses submodularity to recognize that, during any iteration i,

$$f(A_i + \{e\}) - f(A_i) \geq f(A_{i+1} + \{e\}) - f(A_{i+1}),$$

where A_i is the set selected after iteration i and $e \notin A_i$ (Minoux, 1978 [14], Leskovec et al., 2007 [15]). The idea is to maintain a list, sorted in decreasing order, of the marginal increases in f resulting from adding each unselected element. During iteration $i + 1$, we pick the top element of the list and recompute the marginal gain of adding it to the current set, A_i, and re-insert it into the list. If it remains at the top of the list, we add it to A_i to obtain $A_{i+1} = A_i$. As a result, we do not need to evaluate the remaining elements. The second drawback is that, for most diffusion models (including the IC model), computing the influence of a seed set is $\#P$-hard and must be estimated via simulation (Chen et al., 2010 [16]). Several works have proposed various methods for addressing these issues, including Kimura and Saito, 2006 [17], Chen et al., 2009 [18], Goyal et al., 2011 [19], and Mirzasoleiman et al., 2014 [20].

Greedy algorithms can also be applied to finding approximations for the budget allocation problem as well. For their source-side influence model, Alon et al., 2012 [1] obtained a $(1 - 1/e)$-approximation by enumerating all allocations involving three source nodes and greedily completing each initial allocation. Their algorithm is impractical from a runtime perspective as it requires enumerating at least $\mathcal{O}(B^3 |S|^3)$ initial solutions, resulting in $\mathcal{O}(B^5 |S|^4)$ evaluations of the influence function.

Soma et al., 2014 [5] demonstrate that the IC model for budget allocation admits a monotone DR-submodular influence function and prove that the greedy completion algorithm by Alon et al. is applicable for the more general problem of maximizing a monotone lattice submodular function subject to a knapsack constraint. They further demonstrate that the enumeration step is unnecessary when functions are DR-submodular and costs are uniform (which applies in budget allocation), decreasing the number of function evaluations to $\mathcal{O}(B |S|)$, matching the influence maximization case.

The practical runtime of using the greedy algorithm for budget allocation can be similarly reduced by using lazy evaluations (made possible by the diminishing returns property). Moreover, unlike in influence maximization, influence functions are computable in the budget allocation setting, requiring a runtime of $\mathcal{O}(B |S| |T|)$ to compute under the IC model.

2.4.2 Continuous methods

The analogue between submodularity and concavity can be leveraged for maximization by using the *multilinear extension* of a submodular set function. We can view a submodular set function $f : 2^V \to \mathbb{R}_+$ as a function defined over the corners of a unit cube $f : \{0, 1\}^{|V|} \to \mathbb{R}_+$. Naturally, we look to extend f to the entire unit cube, i.e., $F : [0, 1]^{|V|} \to \mathbb{R}_+$.

Definition 2.4.1. (**Multilinear extension**) The *multilinear extension* of f, $F : [0, 1]^{|V|} \to \mathbb{R}_+$, is defined as the expected value of f over all random subsets $A \subseteq V$, denoted $Ra(A)$,

where $\Pr(i \in A) = \mathbf{x}(i)$.

$$F(\mathbf{x}) = \mathbb{E}[f(Ra(A))] = \sum_{A \subseteq V} f(A) \prod_{i \in A} \mathbf{x}(i) \prod_{j \notin A} (1 - \mathbf{x}(j)).$$

For monotone submodular function maximization subject to a general matroid constraint, a $(1 - 1/e)$-approximation can be achieved by using the continuous greedy algorithm to maximize the multilinear extension subject to a matroid polytope constraint, followed by a rounding scheme to recover a discrete solution (Vondrak, 2008 [21]). The continuous greedy algorithm has a few downfalls, primarily that it is impractically slow, requiring $\Theta(|V|^8)$ evaluations of the multilinear extension. Moreover, evaluating the multilinear extension itself is difficult as it requires $2^{|V|}$ queries to f. As a result, it is typically approximated using sampling, which can also become expensive. A large body of work has looked at extensions to as well as accelerated variations of the continuous greedy method, including Chekuri et al., 2009 [22], Chekuri and Vondrak, 2009 [23], Calinescu et al., 2011 [24], Feldman et al., 2011 [25], Badanidiyuru and Vondrak, 2014 [26], and Chekuri et al., 2015 [27].

Chapter 3

Robust Budget Allocation

A critical weakness of the previously discussed methods for budget allocation is the assumption that the diffusion process, and therefore the influence function, is available a priori. Previous work addresses this issue by assuming a diffusion model, then using simplifying assumptions to select edge parameters or estimating edge parameters using historical observations (Saito et al., 2008 [6], Goyal et al., 2011 [19], Netrapalli and Sanghavi., 2012 [7], Gomez-Rodriguez et al., 2012 [28], Gomez-Rodriguez et al., 2014 [8]). Yet methods that assume a diffusion model are generally problematic, since the diffusion model itself may not accurately capture real-life phenomena (Hu et al., 2018 [9]). Other work has looked to estimate the influence function without the assumption of any particular diffusion model to circumvent the issue of model mis-specification (Du et al., 2013 [29], 2014 [30]).

Alternatively, in this thesis, we take a robust optimization perspective to budget allocation, where we assume the influence function can vary arbitrarily within an uncertainty set, capturing both cases of parameter uncertainty and model misspecification. The goal of this problem, known as *robust budget allocation*, is to find a budget allocation that maximizes the worst-case influence function in the uncertainty set.

3.1 Problem formulation

We consider the same setting as defined in Section 2.1. The robust budget allocation problem aims to find a budget allocation that maximizes the worst-case influence function, $\min_{f\in\Sigma} f(\cdot)$, which we refer to as the *robust influence*. Formally, the problem is

$$\max_{\mathbf{x}\in\mathbb{Z}_+^{|S|}} \min_{f\in\Sigma} f(\mathbf{x}) \quad \text{s.t.} \quad c(\mathbf{x}) \leq B, \qquad \mathbf{x} \leq \mathbf{r}, \tag{3.1}$$

where Σ is an uncertainty set of candidate influence functions, $|\Sigma|$ is finite, and all $f \in \Sigma$ are monotone DR-submodular. In other words, we are solving the budget allocation problem subject to an adversarially chosen influence function. The influence functions composing

Σ can arise from any diffusion model, not just limited to those discussed in Section 2.2, or estimation method so long as they are monotone DR-submodular.

Clearly, robust budget allocation is NP-hard since it reduces to standard budget allocation when $|\Sigma| = 1$. The primary challenge is that we cannot directly use previously discussed methods because the minimum of DR-submodular functions is not necessarily DR-submodular. However, we can still leverage the DR-submodular structure of the problem to design an efficient approximation algorithm.

3.1.1 Robust ratio formulation

We note an alternative formulation of the robust problem that involves the *robust ratio*, defined as

$$\rho(\mathbf{x}) = \min_{f \in \Sigma} \frac{f(\mathbf{x})}{f(\mathbf{x}_f^*)}, \tag{3.2}$$

where $\mathbf{x}_f^*$ is an optimal budget allocation that maximizes f (He and Kempe, 2016 [31] and Chen et al., 2016 [32]). The robust ratio variation of robust budget allocation is

$$\max_{\mathbf{x} \in \mathbb{Z}_+^{|S|}} \rho(\mathbf{x}) \quad \text{s.t.} \quad c(\mathbf{x}) \leq B, \quad \mathbf{x} \leq \mathbf{r}. \tag{3.3}$$

The goal of this problem is slightly different than that of Problem 3.1. Here, we wish to ensure that for each influence function, our solution will be close to that function's optimal solution, regardless of which influence function results in the worst-case scenario. Because of the subtle differences, there may exist settings or applications where one objective is more meaningful than the other. Although the results of this thesis are for Problem 3.1, our algorithm holds for the robust ratio problem and an analogous bi-criteria guarantee can be derived. This is discussed in Appendix D.1.

We emphasize that a solution that maximizes the robust ratio does not necessarily result in a solution that maximizes the robust influence (and vice versa). In fact, for influence maximization over set V, Kalimeris et al., 2019 [33] prove that using the robust ratio as a surrogate objective for the robust influence leads to a solution that is up to a multiplicative factor of $\sqrt{|V|}$ worse than the optimal robust solution. A similar result for budget allocation can be obtained via a reduction fromt he DR-submodular setting to the submodular setting.

3.2 Uncertainty sets

There are many ways to form and interpret the uncertainty set of influence functions. Below, we describe a few natural examples. First, as previously mentioned, the uncertainty set can consist of influence functions that arise from different models or are inferred from different algorithms where we are uncertain as to which method is correct.

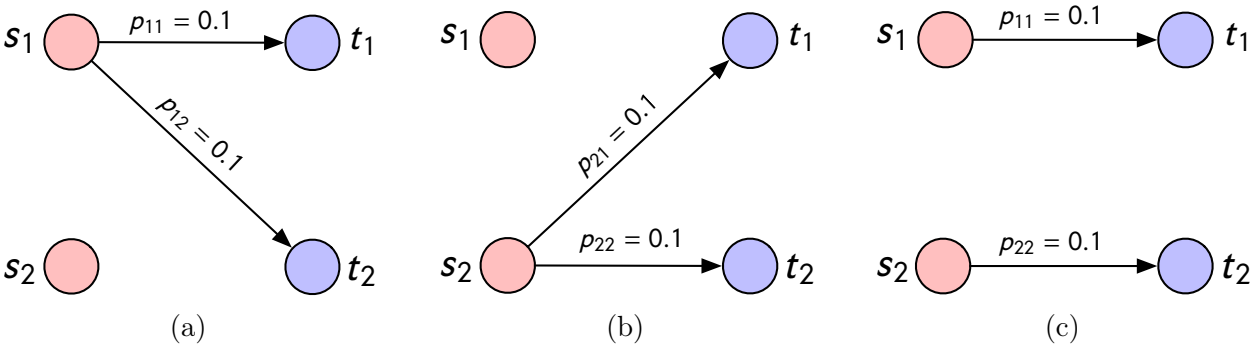


Figure 3.1: An example of an uncertainty set, $\Sigma = \{f_a, f_b, f_c\}$, where each influence function results from a different graph. The influence functions are $f_a(\mathbf{x}) = 2(1 - 0.9^{\mathbf{x}(1)})$, $f_b(\mathbf{x}) = 2(1 - 0.9^{\mathbf{x}(2)})$, and $f_c(\mathbf{x}) = (1 - 0.9^{\mathbf{x}(1)}) + (1 - 0.9^{\mathbf{x}(2)})$.

Different influence functions can also arise from different graphs on the same node set. For example, in the context of marketing, consider a company that wants to place ads across various social media platforms. Although information as to whether potential customers have profiles on these platforms may be available, the company may not know whether the targeted individuals are actively engaging on these platforms and thus not know on which platforms would ads be most effective. Here, the uncertainty set is composed of influence functions arising from different graphs, each representing a different social media platform. An example of this is illustrated in Figure 3.1.

Additionally, the uncertainty set can result from uncertainty about model parameters. This is captured by the *perturbation interval model*, initially proposed for influence maximization by He and Kempe, 2015 [31]. Here, we consider the IC model and assume each edge parameter is only known up to a confidence interval, i.e., $p_{s,t} \in [l_{s,t}, u_{s,t}] \subseteq [0, 1]$, and each different combination of edge parameters form a different influence function. However, due to the continuity of the confidence intervals, Σ is uncountably infinite in this model, violating our requirement that Σ must be finite. This issue can be circumvented by formulating a method to map Σ onto a discrete set.

One possibility is sampling each confidence interval, but this could result in an intractably large Σ. To reduce the number of influence functions, we can look to eliminate dominated influence functions (Krause et al., 2008 [34]).

Definition 3.2.1. An influence function f_i is *dominated* by f_j if for all $\mathbf{0} \leq \mathbf{x} \leq \mathbf{r}$, $f_i(\mathbf{x}) \geq f_j(\mathbf{x})$.

Since dominated influence functions will never be the worst case function chosen by an adversary, we can eliminate them from our uncertainty set. By using this principle, we prove the following lemma:

Lemma 1. Let each $p_{s,t}$, the probability that node u activates node v, lie within a confidence interval $[l_{s,t}, u_{s,t}] \subseteq [0, 1]$. For any budget allocation the worst possible influence is achieved by fixing $p_{s,t} = l_{s,t}$.

The proof of this lemma can be found in Appendix A.

For the robust ratio setting, Lemma 1 may not necessarily hold. However, for the influence maximization problem, He and Kempe, 2016 [35] prove that only the interval endpoints need to be considered, resulting in $|\Sigma| = 2^{|E|}$, where $|E|$ is the number of edges in the graph. Demonstrating an analogous result for the budget allocation case is left as future work.

3.3 Hardness of approximation

Since robust budget allocation is a generalization of budget allocation, it is already known to be NP-hard to obtain an approximation better than $(1 - 1/e)$. Moreover, with the following theorem, we prove that no polynomial-time algorithm is guaranteed to achieve a good approximation unless $P = NP$. The proof of this theorem relies on a reduction to the hitting set problem and can be found in Appendix B.1.

Theorem 2. *Assume $P \neq NP$. Given a function $\gamma(\cdot, \cdot) > 0$, there are no polynomial time algorithms for the following problem: Given n source nodes, m target nodes, a set Σ of influence functions on these nodes, and a cost constraint B, find a budget allocation, $\mathbf{x}$, satisfying the cost constraint $c(\mathbf{x}) \leq B$ such that $\min_{f\in\Sigma} f(\mathbf{x}) \geq \gamma(n, m) \min_{f\in\Sigma} f(\mathbf{x}^*)$, where $\mathbf{x}^*$ is an optimal solution of Problem 3.1.*

As a result, we must either relax the cost constraint or relax the optimization objective to obtain any non-trivial guarantee. Since we use a knapsack constraint, which has a straightforward relaxation, we choose to relax the cost constraint by a multiplicative factor $\eta \geq 1$, resulting in the following problem:

$$\max_{\mathbf{x}\in\mathbb{Z}_+^{|S|}} \min_{f\in\Sigma} f(\mathbf{x}) \quad \text{s.t.} \quad c(\mathbf{x}) \leq \eta B, \quad \mathbf{x} \leq \mathbf{r}. \tag{3.4}$$

3.4 Related work

We discuss previous work that study robustness relating to submodular maximization in general, as well as methods specifically addressing the robust influence maximization and budget allocation problems.

3.4.1 Greedy methods

Krause et al., 2008 [34] introduced the general problem of robust submodular optimization, which looks to solve

$$\max_{A \subseteq V} \min_{f \in \Sigma} f(A) \quad \text{s.t.} \quad |A| \leq k, \tag{3.5}$$

where all $f \in \Sigma$ are monotone submodular. They prove that under $P \neq NP$, unless an algorithm over-selects by a factor of $\alpha > 1$, a polynomial time approximation is not possible. Instead, they look to approximate a relaxation, where the cardinality constraint is replaced with $|A| \leq \alpha k$. They propose the SATURATE algorithm, which reduces the problem to a submodular set cover problem that can be solved using the greedy algorithm. For a sufficiently large α, SATURATE finds a solution, $A \subseteq V$, guaranteed to satisfy

$$\min_{f \in \Sigma} f(A) \geq \max_{|A| \leq k} \min_{i=1,\ldots,m} f(A). \tag{3.6}$$

The approach to robust budget allocation of this thesis builds upon the framework of SATURATE, extending it to functions on the integer lattice.

SATURATE has been extended by many other works including Powers et al. 2016a [36], Powers et al. 2016b [37], and Anari et al., 2019[38]. In particular, He and Kempe, 2016 [35] adapted the algorithm to obtain a bi-criteria approximation algorithm for robust influence maximization using the robust ratio, defined for set functions as

$$\rho(A) = \min_{f \in \Sigma} \frac{f(A)}{f(A_f^*)}, \tag{3.7}$$

where A_f^* is the influence maximizing set for f. They achieve a $(1 - 1/e)$-approximation to the optimal robust ratio value by allowing the cardinality constraint to exceed k by a factor of $\ln(|\Sigma|)$.

Robust influence maximization via the robust ratio is also studied by Chen et al. 2016 [32] using a greedy approach combined with adaptive sampling methods. Kalimeris et al. 2019 [33] examine robust influence maximizing with the view that edge probabilities are functions of user features and a global hyperparameter and optimize over hyperparameter values. Their proposed algorithm uses multiplicative weights updates to generate a sample of influence functions, then performing optimization using the greedy algorithm.

3.4.2 Continuous methods

Robust submodular maximization has also been studied in the context of multi-objective submodular maximization. Chekuri et al., 2009 [22] generalizes the continuous greedy algorithm to obtain a $(1 - 1/e - \epsilon)$-approximation for maximizing a constant number of monotone submodular functions subject to a matroid constraint. Udwani, 2018 [39] extends the work to the case where the number of functions is $\mathcal{O}\left(\frac{k}{(\log k)^3}\right)$.

Staib and Jegelka, 2017 [40] study a continuous version of robust budget allocation, which exploits the fact that influence functions under the IC model are continuous submodular in the edge parameters. However, their method does not consider the additional complication of rounding from a continuous solution to a discrete solution.

Chapter 4

Algorithms and Theoretical Results

In this chapter, we discuss our proposed approach to the robust budget allocation problem. We aim to find an approximation algorithm for a relaxed version, since, by Theorem 2, we have established that non-trivial guarantees cannot be achieved without some relaxation. We present our main algorithm, SaturateDR, which follows the framework of the submodular saturation algorithm (Krause et al., 2008 [34]). SaturateDR relies on the DRPC subroutine, adapted from work by Soma and Yoshida, 2015 [41], to solve the DR-submodular cover problem. Finally, we provide a theoretical analysis of the bi-criteria approximation achieved by SaturateDR.

4.1 Problem reformulation

We begin by re-writing the relaxed robust budget allocation problem (Problem 3.4) as a maximization of a scalar, $\xi > 0$, subject to the existence of an allocation, $\mathbf{x}$, that satisfies $f(\mathbf{x}) \geq \xi$ for all $f \in \Sigma$ as well as the cost and budget cap constraints:

$$\begin{aligned} \max_{\xi, \mathbf{x}} \xi \qquad \text{s.t. } & f(\mathbf{x}) \geq \xi, \forall f \in \Sigma \\ & c(\mathbf{x}) \leq \eta B \\ & \mathbf{x} \leq \mathbf{r}. \end{aligned} \tag{4.1}$$

We can combine the constraints $f(\mathbf{x}) \geq \xi$ for all $f \in \Sigma$ by defining the function

$$H^{(\xi)}(\mathbf{x}) := \sum_{f \in \Sigma} h_f^{(\xi)}(\mathbf{x}), \tag{4.2}$$

where

$$h_f^{(\xi)}(\mathbf{x}) := \min\{\xi, f(\mathbf{x})\}, \tag{4.3}$$

and ξ is a given constant. Observe that $f(\mathbf{x}) \geq \xi$ for all $f \in \Sigma$ if and only if $H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\, \xi$. This can be seen as follows: if $f(\mathbf{x}) \geq \xi$ for all $f \in \Sigma$, then $h_f^{(\xi)}(\mathbf{x}) = \xi \geq \xi$ for all $f \in \Sigma$, and

$H^{(\xi)}(\mathbf{x}) = |\Sigma|\,\xi \geq |\Sigma|\,\xi$. Conversely, if $H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi$, it must be that $H^{(\xi)}(\mathbf{x}) = |\Sigma|\,\xi$, since it can at most equal $|\Sigma|\,\xi$. As a result, $h_f^{(\xi)} = \xi$ for all $f \in \Sigma$, meaning $f(\mathbf{x}) \geq \xi$ for all $f \in \Sigma$.

We re-write Problem 4.1 using function $H^{(\xi)(\cdot)}$:

$$\begin{aligned} \max_{\xi,\mathbf{x}} \xi \qquad & \text{s.t. } H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi \\ & c(\mathbf{x}) \leq \eta B \\ & \mathbf{x} \leq \mathbf{r}. \end{aligned} \tag{4.4}$$

4.2 Algorithm overview and theoretical guarantees

SaturateDR satisfies the folllowing bi-criteria guarantee:

Theorem 3. *Given a fixed budget B and choosing an update size $\delta < \gamma/(2\min_{f\in\Sigma} f(\mathbf{r}))$,* SaturateDR *finds an approximate solution $\hat{\mathbf{x}}$ to the robust budget allocation problem satisfying*

$$\min_{f\in\Sigma} f(\hat{\mathbf{x}}) \geq (1-\delta)\left(\min_{f\in\Sigma} f(\mathbf{x}^*) - \gamma\right) \tag{4.5}$$

$$c(\hat{\mathbf{x}}) \leq (1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right) B, \tag{4.6}$$

where $\mathbf{x}^*$ is the optimal solution to robust budget allocation, $0 < \epsilon, \gamma < 1$ are approximation parameters, $d_{\max} = \max_{s\in S} H^{(\xi)}(\chi_s)$, and $\beta_{\min} = \min\{H^{(\xi)}(\chi_s|\mathbf{x}) : s \in S, \mathbf{x} \in \mathbb{Z}_+^S, H^{(\xi)}(\chi_s|\mathbf{x}) > 0\}$, where $\xi = \min_{f\in\Sigma} f(\mathbf{r})$.

The proof is given in Section 4.4.

SaturateDR find a solution by iteratively evaluating two steps. First, for a given ξ, we find a minimum cost allocation such that $H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi$ is satisfied. We note that $H^{(\xi)}(\mathbf{x})$ is monotone DR-submodular because monotonicity and DR-submodularity are preserved under truncations and summations. As a result, this problem,

$$\begin{aligned} \arg\min_{\mathbf{x}} c(\mathbf{x}) \qquad & \text{s.t. } H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi \\ & \mathbf{x} \leq \mathbf{r}, \end{aligned} \tag{4.7}$$

is an instance of the DR-submodular cover problem, which we solve using the DRPC subroutine, an adaption of the greedy-style decreasing threshold algorithm by Soma and Yoshida, 2015 [41]. DRPC guarantees finding an allocation satisfying the bi-criteria guarantee below. The proof, which follows directly from Soma and Yoshida, can be found in Appendix C.

Theorem 4. *Given a coverage parameter $\xi > 0$, a monotone DR-submodular function $H^{(\xi)}(\mathbf{x})$, and an additive cost function $c(\mathbf{x})$,* DRPC *returns an allocation $\mathbf{x}$ satisfying*

$$H^{(\xi)}(\mathbf{x}) \geq (1-\delta)\,|\Sigma|\,\xi \tag{4.8}$$

$$c(\mathbf{x}) \leq (1+3\epsilon)\left(1+\log\left(\frac{d}{\beta}\right)\right)c(\mathbf{x}^*), \tag{4.9}$$

where $\mathbf{x}^*$ is the optimal minimum cost solution satisfying $H^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi$, $0 < \epsilon, \delta < 1$ are algorithm parameters, $d = \max_{s\in S} H^{(\xi)}(\chi_s)$ is the maximum value of the objective over unit vectors χ_s, and $\beta = \min\{H^{(\xi)}(\chi_s|\mathbf{x}) : s \in S, \mathbf{x} \in \mathbb{Z}_+^S, H^{(\xi)}(\chi_s|\mathbf{x}) > 0\}$ is the minimum marginal increase in the objective over the feasible region.

Second, we update ξ based on whether the resulting allocation of DRPC satisfies the relaxed cost constraint, $c(\mathbf{x}) \leq \eta B$. The relaxation factor η is set using the result of Theorem 4. We upper bound the sub-optimality factor of the cost constraint with a quantity that is independent of ξ. This can be achieved by letting $\xi = \min_{f\in\Sigma} f(\mathbf{r})$ and defining $d_{\max} := \max_{s\in S} H^{(\xi)}(\chi_s)$ and $\beta_{\min} := \min\{H^{(\xi)}(\chi_s|\mathbf{x}) : s \in S, \mathbf{x} \in \mathbb{Z}_+^S, H^{(\xi)}(\chi_s|\mathbf{x}) > 0\}$. The resulting relaxation factor is

$$\eta = (1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right). \tag{4.10}$$

Repeating these two steps until ξ converges, we obtain an allocation, $\hat{\mathbf{x}}$, that satisfies the guarantee of Theorem 3.

4.3 Hardness of bi-criteria approximation

The following theorem demonstrates that our choice of η cannot be improved. In other words, if we want a polynomial-time algorithm that returns an allocation that satisfies the lower bound on robust influence given in the guarantee of SATURATEDR (Equation 4.5), η cannot be less than $(1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right)$. The proof can be found in Appendix B.2.

Theorem 5. *Assume $P \neq NP$. Let $\xi = \min_{f\in\Sigma} f(\mathbf{r})$ and define $d_{\max} := \max_{s\in S} H^{(\xi)}(\chi_s)$ and $\beta_{\min} := \min\{H^{(\xi)}(\chi_s|\mathbf{x}) : s \in S, \mathbf{x} \in \mathbb{Z}_+^S, H^{(\xi)}(\chi_s|\mathbf{x}) > 0\}$, where $H^{(\xi)}(\cdot) := \sum_{f\in\Sigma} \min\{\xi, f(\cdot)\}$. Let $0 < \delta, \epsilon, \gamma < 1$ be approximation parameters. For any budget $B \in \mathbb{Z}_+$, there does not exist any polynomial-time algorithm that is guaranteed to find a budget allocation satisfying*

$$\min_{f\in\Sigma} f(\mathbf{x}) \geq (1-\delta)\left(\min_{f\in\Sigma} f(\mathbf{x}^*) - \gamma\right) \tag{4.11}$$

$$c(\mathbf{x}) \leq \eta B, \tag{4.12}$$

where $\eta \leq (1-\alpha)(1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right)$, for some fixed $\alpha > 0$.

4.4 SaturateDR algorithm

SaturateDR, given in Algorithm 2, takes as input the uncertainty set of influence functions Σ, a budget B, and approximation parameters $0 < \gamma, \epsilon, \delta < 1$. Using a binary search, SaturateDR finds the maximum ξ such that there exists a budget allocation where all influence functions $f \in \Sigma$ achieve a coverage level of at least ξ and the relaxed cost constraint is satisfied.

The upper and lower bounds on ξ are initialized as $\xi_{\min} = 0$ and $\xi_{\max} = \min_{f\in\Sigma} f(\mathbf{r})$. In each iteration, we run the DRPC subroutine using coverage level $\xi = (\xi_{\min} + \xi_{\max})/2$. The resulting $\mathbf{x}$ will fall under two cases:

(i) If the relaxed cost constraint is not satisfied (i.e., $c(\mathbf{x}) > (1+3\epsilon)\left(1+\log\left(\frac{d}{\beta}\right)\right)B$), ξ is an upper bound on the coverage level and we set $\xi_{\max} = \xi$. Due to the monotonicity of $H^{(\xi)}(\mathbf{x})$ and $c(\mathbf{x})$, if we want to achieve a higher coverage level, we must use a higher cost allocation. Since the cost of $\mathbf{x}$ has already exceeded the cost constraint, we will not be able to achieve a higher coverage level.

(ii) If the relaxed cost constraint is satisfied, we retain the resulting allocation as our best current allocation. From Theorem 4, we know that $\mathbf{x}$ satisfies $H^{(\xi)}(\mathbf{x}) \geq (1-\delta)\,|\Sigma|\,\xi$, so we know $\min_{f\in\Sigma} f(\mathbf{x}) \geq (1-\delta)\xi$ and we set $\xi_{\min} = (1-\delta)\xi$.

We repeat this process until $\xi_{\min}$ and $\xi_{\max}$ converges within a threshold γ. The resulting allocation satisfies the guarantee in Theorem 3.

Algorithm 2 SaturateDR$(\Sigma, B, \gamma, d_{\max}, \beta_{\min})$

Initialize $\xi_{\min} \leftarrow 0, \xi_{\max} \leftarrow \min_{f\in\Sigma} f(\mathbf{r})$
while $(\xi_{\max} - \xi_{\min}) \geq \gamma$ **do**
  $\xi \leftarrow (\xi_{\max} + \xi_{\min})/2$
  Define $H^{(\xi)}(\mathbf{x}) \leftarrow \sum_{f\in\Sigma} h_f^{(\xi)}(\mathbf{x})$
  $\mathbf{x} \leftarrow \text{DRPC}(H^{(\xi)}(\mathbf{x}), |\Sigma|\xi, \epsilon, \gamma)$
  if $\sum_{s\in S} \mathbf{x}(s) > (1+3\epsilon)(1+\log(d_{\max}/\beta_{\min}))B$ **then**
    $\xi_{\max} \leftarrow \xi$
  else
    $\hat{\mathbf{x}} \leftarrow \mathbf{x}$
    $\xi_{\min} \leftarrow (1-\delta)\xi$
return $\hat{\mathbf{x}}$

We note that with slight modifications, SaturateDR can also be used for the robust ratio version of robust budget allocation (Problem 3.3). Moreover, an analogous guarantee can be proved, which is presented in Appendix D.1.

4.4.1 Proof of Theorem 3

Proof. The budget allocation, $\hat{\mathbf{x}}$, resulting from SaturateDR, is the output of running DRPC using the optimal ξ, which we denote as ξ^*. Thus, $\hat{\mathbf{x}}$ must satisfy Theorem 4:

$$H^{(\xi^*)}(\hat{\mathbf{x}}) \geq (1-\delta)\,|\Sigma|\,\xi^* \tag{4.13}$$

$$c(\hat{\mathbf{x}}) \leq (1+3\epsilon)\left(1+\log\left(\frac{d}{\beta}\right)\right)c(\mathbf{x}^*). \tag{4.14}$$

We immediately obtain our cost guarantee from Equation 4.14. To obtain our objective guarantee, we note that Equation 4.13 is equivalent to

$$\min_{f\in\Sigma} f(\hat{\mathbf{x}}) \geq (1-\delta)\xi^*.$$

Recall that SaturateDR terminates when $\xi_{\max} - \xi_{\min} < \gamma$. Since $\xi_{\min} = \xi^*$, we have $\xi^* \geq \xi_{\max} - \gamma$ during the last iteration. Additionally, we have that $\xi_{\max} \geq \min_{f\in\Sigma} f(\mathbf{x}^*)$, because $\xi_{\max}$ is an upper bound on possible ξ's. Using these inequalities, we obtain our objective guarantee:

$$\begin{aligned}
\min_{f\in\Sigma} f(\hat{\mathbf{x}}) &\geq (1-\delta)\xi^* \\
&\geq (1-\delta)(\xi_{\max}-\gamma) \\
&\geq (1-\delta)(\min_{f\in\Sigma} f(\mathbf{x}^*)-\gamma).
\end{aligned}$$

It remains to show that the binary search for the optimal ξ terminates in a finite number of iterations. We ignore the case where $\xi_{\max} \leftarrow \xi$, because this is the update definition for a standard binary search, which we know will terminate. Consider the case where only $\xi_{\min}$ is updated. Define $\Delta^{(i)} := \xi^{(i)}_{\max} - \xi^{(i)}_{\min}$. Since we set $\xi_{\min} = (1-\delta)\xi$ rather than $\xi_{\min} = \xi$, we need to show that $\xi^{(i+1)}_{\max} - \xi^{(i+1)}_{\min} < \Delta^{(i)}$:

$$\begin{aligned}
\xi^{(i+1)}_{\max} - \xi^{(i+1)}_{\min} &= \xi^{(i)}_{\max} - \xi^{(i+1)}_{\min} \\
&= (\xi^{(i)}_{\max} - \xi^{(i)}) + (\xi^{(i)} - \xi^{(i+1)}_{\min}) \\
&= \frac{\Delta^{(i)}}{2} + \xi^{(i)} - \xi^{(i+1)}_{\min} \\
&= \frac{\Delta^{(i)}}{2} + \xi^{(i)} - (1-\delta)\xi^{(i)} \\
&= \frac{\Delta^{(i)}}{2} + \delta\xi^{(i)} \\
&\leq \frac{\Delta^{(i)}}{2} + \delta\xi^{(i)}_{\max} \\
&\leq \frac{\Delta^{(i)}}{2} + \delta\left(\min_{f\in\Sigma} f(\mathbf{r})\right).
\end{aligned}$$

Choosing $\delta < \gamma/(2\min_{f\in\Sigma} f(\mathbf{r}))$, we obtain

$$\xi_{\max}^{(i)} - \xi_{\min}^{(i+1)} < \frac{\Delta^{(i)}}{2} + \frac{\gamma}{2}.$$

Since $\Delta^{(i)} \geq \gamma$ for iteration $i+1$ to occur, we have that

$$\xi_{\max}^{(i)} - \xi_{\min}^{(i+1)} < \Delta^{(i)}.$$

As a result, the binary search will terminate in $\mathcal{O}(1/\gamma)$ iterations. □

4.5 DRPC algorithm

The DRPC subroutine solves Problem 4.7, which is an instance of the DR-submodular cover problem. Although it is an NP-hard problem, Soma and Yoshida, 2015 [41] provide a greedy-style polynomial-time bi-criteria algorithm. We formulate DRPC by making the following adaptations to Soma and Yoshida's algorithm:

(i) We fix the cost function to be additive: $c(\mathbf{x}) = \sum_{s\in S} \mathbf{x}(s)$. As a result, $c(\chi_s) = 1$ for all $s \in S$ and the curvature term is fixed to be $\rho = 1$.

(ii) We allow the nodal budget capacities, $\mathbf{r}$, to vary based on the node. In doing so, we modify the lower bound of the threshold to be $\frac{\delta d}{n r_{\max}}$, where $r_{\max} = \max_{s\in S} \mathbf{r}(s)$.

DRPC takes as input a monotone, DR-submodular function, $f(\cdot)$, a coverage parameter, $\xi > 0$, and approximation parameters, $0 < \epsilon, \delta < 1$. We take $H^{(\xi)}(\cdot)$ as our input function $f(\cdot)$. DRPC initializes an empty allocation, $\mathbf{x} = \mathbf{0}$, and greedily adds elements to each coordinate of $\mathbf{x}$ such that the marginal gain-to-cost ratio, $\frac{f(k\chi_s|\mathbf{x})}{k}$, is above a threshold θ. After iterating through all coordinates of $\mathbf{x}$, the threshold θ is decreased and this process is repeated until either the objective meets the desired coverage level, $|\Sigma|\,\xi$, or θ reaches its lower bound, $\frac{\delta d}{n r_{\max}}$. DRPC guarantees that the coverage will be met up to at least $(1-\delta)\,|\Sigma|\,\xi$.

Algorithm 3 DRPC$(f, \xi, \epsilon, \delta)$

Initialize $\mathbf{x} \leftarrow \mathbf{0}$, $d \leftarrow \max_{s\in S} f(\chi_s)$, $r_{\max} \leftarrow \max_{s\in S} \mathbf{r}(s)$, $n \leftarrow |S|$
for $\theta = d; \theta \geq \frac{\delta d}{n r_{\max}}; \theta \leftarrow \theta(1-\epsilon)$ **do**
    for $s \in S$ **do**
        Find maximum integer $0 < k \leq \mathbf{r}(s) - \mathbf{x}(s)$ s.t. $\frac{f(k\chi_s|\mathbf{x})}{k} \geq \theta$ using a binary search
        if k exists **then** $\mathbf{x} \leftarrow \mathbf{x} + k\chi_s$
        if $f(\mathbf{x}) \geq \xi$ **then** break the outer for loop
return $\mathbf{x}$

4.6 Runtime and scalability

Let θ_H denote the runtime of an evaluation oracle for function $H^{(\xi)}(\cdot)$. The runtime of SATURATEDR is $\mathcal{O}\left(\frac{1}{\epsilon} n \log(\frac{nr_{\min}}{\delta})(\theta_H \log(r_{\max}) + \theta_H) \log(\frac{1}{\gamma})\right)$, where $0 < \gamma, \epsilon, \delta < 1$ are algorithm parameters. We note that the bulk of the runtime comes from DRPC, which has a runtime of $\mathcal{O}\left(\frac{1}{\epsilon} n \log(\frac{nr_{\min}}{\delta})(\log(r_{\max})\theta_H + \theta_H)\right)$. The runtime of an evaluation oracle for $H^{(\xi)}(\cdot)$ is $\mathcal{O}(|\Sigma|\, \theta_f)$, where θ_f is the runtime of an evaluation oracle for the influence function, $f(\cdot)$. For influence functions under the IC model, $\theta_f = \mathcal{O}(|S|\,|T|)$.

Our runtime is comparable to that of other algorithms for robust budget allocation, namely that of Staib and Jegelka, 2017 [40], which has a runtime of $\mathcal{O}(\frac{1}{\epsilon\delta^3\alpha} n^2 m^2 \log \frac{n}{\delta})$ where ϵ, δ, α are algorithm parameters. Under the IC model, both algorithms have a quadratic dependence on the number of source nodes, but SATURATEDR has only a linear dependence on the number of target nodes. On the other hand, SATURATEDR scales linearly with the size of the uncertainty set whereas Staib and Jegelka's algorithm does not, primarily due to differences in how the uncertainty sets are constructed.

For large graphs or large uncertainty sets, the runtime of DRPC can become fairly expensive due the number of evaluations of $H^{(\xi)}(\cdot)$ as well as the cost of the evaluation itself. To address the former problem, we can exploit the DR-submodularity of $H^{(\xi)}(\cdot)$ to use lazy evaluations (as discussed in Section 2.4.1).

Chapter 5

Simulations

In this Chapter, we verify our theoretical results by implementing SaturateDR for instances of robust budget allocation on synthetic data. We compare the robust influence of allocations obtained by SaturateDR to that of simple heuristics. We compare our solution to two simple heuristics:

(i) Greedy, which is a standard greedy algorithm applied to the objective $\min_{f\in\Sigma} f(\mathbf{x})$. Since the minimum of monotone DR-submodular functions is not necessarily monotone DR-submodular, Greedy can perform arbitrarily badly.

(ii) AllGreedy, which uses a standard greedy solution separately on each $f \in \Sigma$ and chooses the best of the resulting solutions.

5.1 Varying uncertainty set size

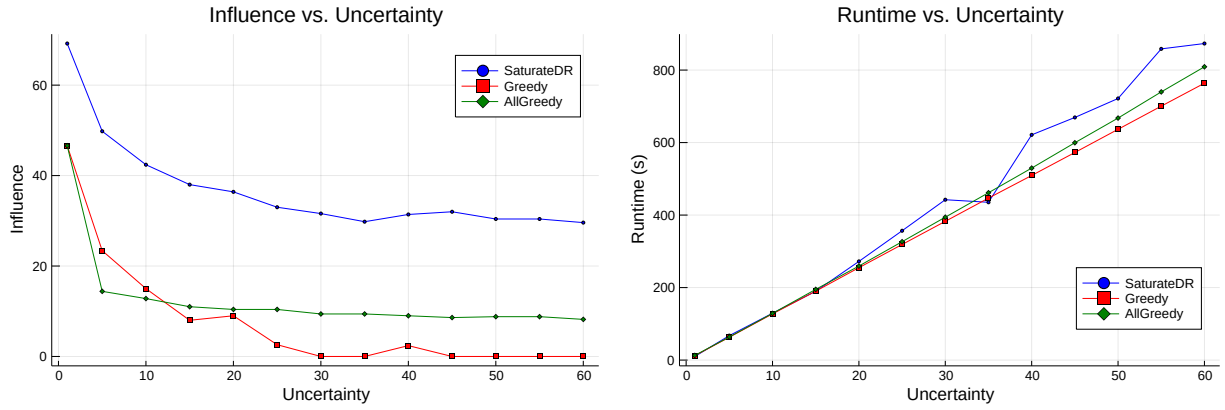
We explore whether our robust solution achieves a gain in robust influence, $\min_{f\in\Sigma} f(\mathbf{x})$, as opposed to the above non-robust heuristics as we vary the size of our uncertainty set. We expect that for a larger uncertainty set, SaturateDR will perform significantly better than the non-robust heuristics.

We look at the performance of SaturateDR and the heuristics described above as the number of functions in the uncertainty set varies. We use randomly generated bipartite graphs of size $|S| = 50$ and $|T| = 1000$. We fix the degree of $s \in S$ to obey a power law of $\gamma = 2$, meaning $d^{-\gamma}$ fraction of randomly chosen source nodes will have degree d. For each $s \in S$ with degree d, we choose d target nodes uniformly at random and connect them to s with edge parameter $p_{s,t} = 1$. We set the budget cap of each node be 1. We set parameters $\epsilon = 0.01$ and $\delta = 0.01$. We consider uncertainty set sizes $|\Sigma| = 1, 5, 10, 15, \ldots 60$.

In Figures 5.1 and 5.2, we see the robust influence, $\min_{f\in\Sigma} f(\mathbf{x})$, plotted against the uncertainty set size, for SaturateDR (in blue), Greedy (in red), and AllGreedy (in green) when using a smaller budget of $B = 10$ and a larger budget $B = 25$, respectively.

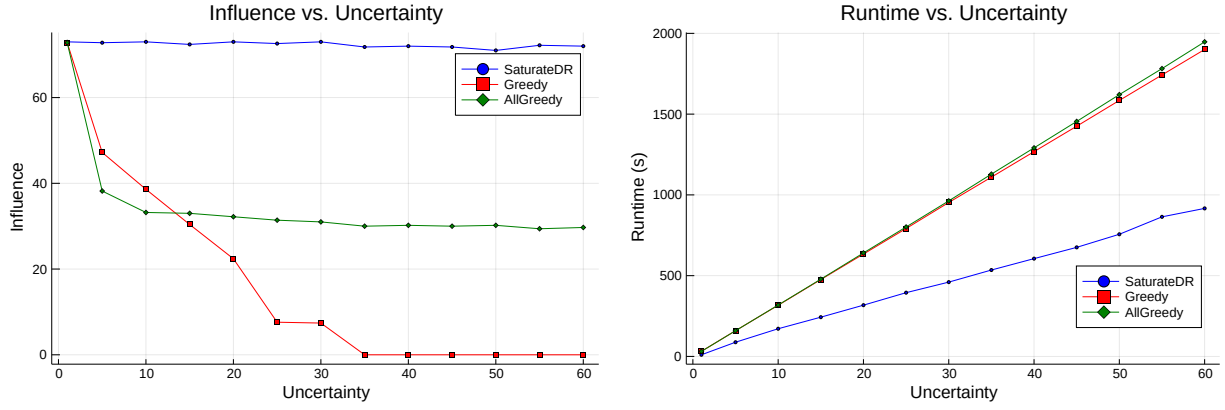
For both budget sizes, SATURATEDR does consistently better than both heuristics for all uncertainty set sizes. We note that on average, SATURATEDR over-selects the budget by a factor of 2.3 for $B = 10$ and 1.9 for $B = 25$, which satisfies the bound of Theorem 3. We note that this over-selection may account for part of the influence gain SATURATEDR has over the heuristic algorithms, however allowing the heuristic algorithms this same over-selection will substantially increase the runtime.

We see that for the smaller budget, SATURATEDR has a comparable runtime to the heuristics, whereas for the larger budget, SATURATEDR has a runtime that is, on average, twice as fast.



(a) (b)

Figure 5.1: Plots showing the performance of SATURATEDR, GREEDY, and ALLGREEDY with budget $B = 10$ for varying uncertainty set sizes. Each point represents an average over 25 trials.



(a) (b)

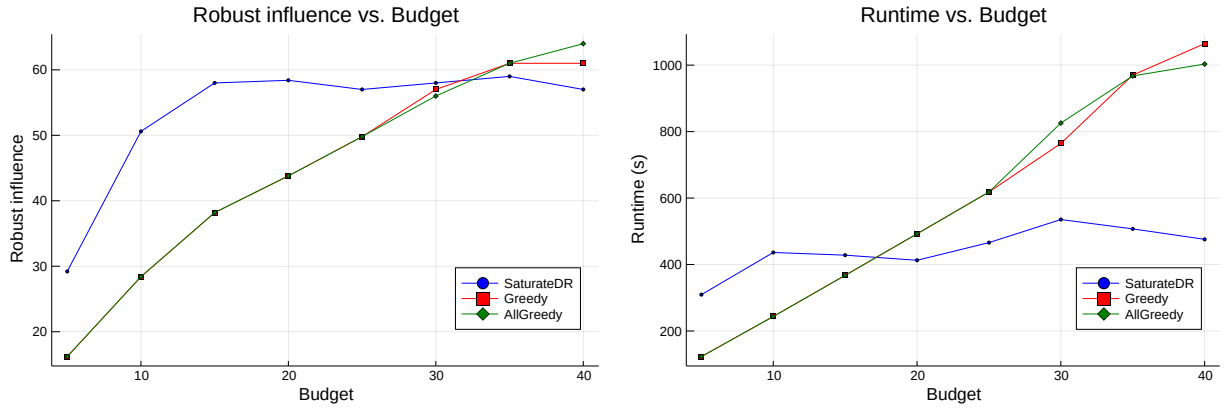
Figure 5.2: Plots showing the performance of SATURATEDR, GREEDY, and ALLGREEDY with budget $B = 25$ for varying uncertainty set sizes. Each point represents an average over 25 trials.

5.2 Varying budget

Here, we explore how changing the budget available affects the robust influence and runtime of SaturateDR as compared to the heuristic algorithms. We explore whether our robust solution achieves a gain in robust influence, $\min_{f \in \Sigma} f(\mathbf{x})$, as opposed to the above non-robust heuristics as we vary the size of our uncertainty set. We expect that for a larger uncertainty set, SaturateDR will perform significantly better than the non-robust heuristics.

We compare SaturateDR to the two heuristics while varying the total budget available. We randomly generate bipartite graphs of size $|S| = 50$ and $|T| = 1000$ using the same method as in the previous section. Here, we fix the edge parameter to be $p_{s,t} = 1$, the uncertainty set size to be $|\Sigma| = 20$, and the budget cap of each node is 1. We consider budgets $B = 5, 10, 15, \ldots 40$.

We can see in Figure 5.3a that when the available budget is small, SaturateDR outperforms both Greedy and AllGreedy. As the available budget increase, the robust influence obtained by the heuristic algorithms increases, surpassing that of SaturateDR when the budget becomes large enough. This implies that robust methods are more necessitated when resources are scarce as opposed to when resources are abundant. Figure 5.3b shows that the runtime of SaturateDR remains fairly consistent regardless of the total budget. On the other hand, the runtime of both heuristics increase linearly as total budget increases.



(a) (b)

Figure 5.3: Plots showing the performance of SaturateDR, Greedy, and AllGreedy for $|\Sigma| = 20$ and varying total budgets. Each point represents an average over 25 trials.

Chapter 6

Conclusions and Future Work

In this thesis, we address the issue of parameter and model uncertainty in the budget allocation problem by studying a robust formulation, where the goal is to find a budget allocation that maximizes influence subject to an adversarially chosen influence function. We extend ideas from robust submodular maximization to the DR-submodular setting to formulate a polynomial-time algorithm that achieves a bi-criteria approximation. We evaluate our algorithm, SaturateDR, on synthetic data, demonstrating the necessity of robust solutions for budget allocation.

Future work includes improving the scalability of our algorithm and evaluating our algorithm on real-life data. Moreover, it would be interesting to further examine the differences between methods optimizing the robust ratio as opposed to the worst-case influence as well as generalize our method in a computationally tractable way to apply to general, non-bipartite graphs.

Appendix A

Proof of Lemma 1

Proof. Given a graph $(S, T; E)$, choose an arbitrary edge $(s', t') \in E$. We assume a fixed budget allocation, $\mathbf{x}$, and fixed probability parameters, $p_{s,t}$, for all edges $(s, t) \in E$ except for edge (s', t'). Let $y := p_{s',t'}$ and allow it to vary within uncertainty set $[l, u] \subseteq [0, 1]$. Let the influence achieved by $\mathbf{x}$ be denoted as $f(\mathbf{x}; y)$.

We write the influence function using the equivalent triggering set formulation. Recall that each $G \in \mathcal{G}$ is a fixed selection of triggering sets and corresponds to a graph where edges are either live or blocked. We write the influence function as

$$f(\mathbf{x}; y) = \sum_{G \in \mathcal{G}} \Pr(G) f_G(\mathbf{x}).$$

The probability $\Pr(G)$ that graph G occurs is

$$\Pr(G) = \begin{cases} (1 - (1 - y)^{\mathbf{x}(s')})\kappa(G) & \text{if } (s', t') \in G \\ (1 - y)^{\mathbf{x}(s')}\kappa(G) & \text{if } (s', t') \notin G, \end{cases}$$

where

$$\kappa(G) = \prod_{(s,t) \in G, (s,t) \neq (s',t')} (1 - (1 - p_{s,t})^{\mathbf{x}(s)}) \prod_{(s,t) \notin G, (s,t) \neq (s',t')} (1 - p_{s,t})^{\mathbf{x}(s)}$$

and number of target nodes activated by $\mathbf{x}$ under graph G is denoted as $f_G(\mathbf{x})$. Let let $\tilde{G}$ be a subset of graph G that excludes consideration of edge (s', t'). We see that each $\tilde{G} \in \tilde{\mathcal{G}}$ corresponds to two graphs in $\mathcal{G}$, one where edge (s', t') is live and one where edge (s', t') is blocked and the remaining edges are the the same as in $\tilde{G}$. We denote these two graphs as $\tilde{G}(s', t')$ and $\tilde{G}(\overline{s', t'})$, respectively.

Given $\mathbf{x}$ where $\mathbf{x}(s') = 0$, the influence function is the following constant

$$\begin{aligned} f(\mathbf{x}; y) &= \sum_{\tilde{G} \in \tilde{\mathcal{G}}} (1 - (1-y)^{\mathbf{x}(s')}) \kappa(\tilde{G}) f_{\tilde{G}(s',t')}(\mathbf{x}) + (1-y)^{\mathbf{x}(s')} \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= \sum_{\tilde{G} \in \tilde{\mathcal{G}}} \left[(1 - (1-y)^{\mathbf{x}(s')}) + (1-y)^{\mathbf{x}(s')} \right] \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= \sum_{\tilde{G} \in \tilde{\mathcal{G}}} \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}). \end{aligned}$$

As a result, an adversary cannot affect the influence achieved with $\mathbf{x}$ by manipulating y.

Given $\mathbf{x}$ where $\mathbf{x}(s') > 0$, each $\tilde{G} \in \tilde{\mathcal{G}}$ will fall under one of the following two cases:

(i) Target node t' will be activated if edge (s', t') is live but will not be activated if it is blocked. In this case, $f_{\tilde{G}(s',t')}(\mathbf{x}) = f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) + 1$. The influence function is

$$\begin{aligned} \Pr(\tilde{G}) f_{\tilde{G}}(\mathbf{x}) &= (1 - (1-y)^{\mathbf{x}(s')}) \kappa(\tilde{G}) f_{\tilde{G}(s',t')}(\mathbf{x}) + (1-y)^{\mathbf{x}(s')} \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= (1 - (1-y)^{\mathbf{x}(s')}) \kappa(\tilde{G}) (f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) + 1) + (1-y)^{\mathbf{x}(s')} \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= \kappa(\tilde{G}) (f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) + 1) - (1-y)^{\mathbf{x}(s')} \kappa(\tilde{G}). \end{aligned}$$

We see that the influence function is monotone increasing in y on the interval $y \in [0, 1]$. By setting y to its lower endpoint, an adversary can minimize the performance of allocation $\mathbf{x}$.

(ii) Target node t' is already activated by another source node, so $f_{\tilde{G}(s',t')}(\mathbf{x}) = f_{\tilde{G}(\overline{s',t'})}(\mathbf{x})$. The influence function is

$$\begin{aligned} \Pr(\tilde{G}) f_{\tilde{G}}(\mathbf{x}) &= (1 - (1-y)^{\mathbf{x}(s')}) \kappa(\tilde{G}) f_{\tilde{G}(s',t')}(\mathbf{x}) + (1-y)^{\mathbf{x}(s')} \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= \left[(1 - (1-y)^{\mathbf{x}(s')}) + (1-y)^{\mathbf{x}(s')} \right] \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}) \\ &= \kappa(\tilde{G}) f_{\tilde{G}(\overline{s',t'})}(\mathbf{x}), \end{aligned}$$

which we can see is constant with respect to y. Once again, an adversary cannot affect the influence achieved with $\mathbf{x}$ by manipulating y.

Repeating this argument for all edges in E, we can see that an adversary can minimize the performance of any budget allocation by setting all edge probability parameters to their lower endpoint. □

Appendix B

Hardness of Approximation Proofs

We prove both Theorem 2 and Theorem 5 using a reduction to the hitting set problem. Consider the following hitting set instance: Let $V = \{a_1, \ldots, a_n\}$ be a ground set of elements, $\mathcal{C} = \{C_1, \ldots, C_m\}$ be a collection of subsets of V, and $B \in \mathbb{Z}_+$ be the cardinality constraint on the hitting set. The goal is to find a hitting set, $A \subset V$, satisfying $|A| \leq B$ such that every subset in $\mathcal{C}$ contains at least one element from A.

We construct an instance of the robust budget allocation problem: Let $S = \{s_1, \ldots, s_n\}$ be a set of source nodes where $s_i \in S$ corresponds to $a_i \in V$. Let $T = \{t_1, \ldots, t_m\}$ be a set of target nodes where $t_j \in T$ corresponds to $C_j \in \mathcal{C}$. Let there be m graphs on this vertex set. For the j-th graph, if $a_i \in C_j$, we place an edge between source node s_i and target node t_j with edge parameter $p_{i,j} = 1$. Each of the $j = 1, \ldots, m$ graphs admits a different influence function, f_j, so we have $|\Sigma| = m$. The cost constraint of the allocation corresponds to the cardinality constraint, making the cost constraint $c(\mathbf{x}) = \sum_{\mathbf{s} \in \mathbf{S}} \mathbf{x}(s) \leq B$. The nodal budget constraint, $\mathbf{r}$, is the all ones vector since in the hitting set problem, we can only choose elements once. Each set, A, corresponds to a budget allocation, $\mathbf{x}$, where $\mathbf{x}(i) = 1$ if $a_i \in A$.

B.1 Proof of Theorem 2

Proof. Suppose set A is an optimal hitting set, satisfying cardinality constraint $|A| \leq B$. Set A corresponds to an $\mathbf{x}$ satisfying $c(\mathbf{x}) \leq B$ and $\mathbf{x} \leq \mathbf{r}$. As a hitting set, A intersects every set in $\mathcal{C}$, meaning for all graphs $j = 1, \ldots, m$, $\mathbf{x}$ will activate target node t_j in the j-th graph. As a result, $\mathbf{x}$ achieves a influence of $\min_{j \in 1, \ldots, m} f_j(\mathbf{x}) = 1$, which is optimal based on our graph construction. If set A is not a hitting set, there must exist some subset C_j that is not intersected. As a result, the corresponding budget allocation, $\mathbf{x}$, cannot activate any target nodes in the j-th graph instance, making the robust influence $\min_{j \in 1, \ldots, m} f_j(\mathbf{x}) = 0$.

Suppose towards contradiction that there exist a polynomial-time algorithm that can solve robust budget allocation with an approximation factor of $\gamma(n, m)$. The algorithm should

select an $\mathbf{x}$ satisfying $c(\mathbf{x}) \leq B$, $\mathbf{x} \leq \mathbf{r}$, and

$$\min_{j\in 1,\dots,m} f_j(\mathbf{x}) \geq \gamma(n,m) \min_{j\in 1,\dots,m} f_j(\mathbf{x}^*), \tag{B.1}$$

where $\mathbf{x}^*$ is an optimal allocation. Since, in this instance, an optimal allocation would achieve a robust influence of 1, Equation B.1 becomes

$$\min_{j\in 1,\dots,m} f_j(\mathbf{x}) \geq \gamma(n,m). \tag{B.2}$$

The allocation selected by the algorithm must either correspond to a hitting set, achieving a robust influence of $\min_{j\in 1,\dots,m} f_j(\mathbf{x}) = 1$, or not correspond to a hitting set, achieving a robust influence of $\min_{j\in 1,\dots,m} f_j(\mathbf{x}) = 0$. Since $\gamma(n,m) > 0$, Equation B.2 can only be satisfied if the algorithm selects an $\mathbf{x}$ with robust influence $\min_{j\in 1,\dots,m} f_j(\mathbf{x}) = 1$, which corresponds to finding a hitting set. This contradicts the NP-hardness of the hitting problem and thus there cannot exists such an algorithm.

□

B.2 Proof of Theorem 5

Proof. Assuming $P \neq NP$, for every $\alpha > 0$, the hitting set problem cannot be approximated within a ratio of $(1-\alpha)\log m$, where $m = |\mathcal{C}|$ (Dinur and Steurer, 2013, Corollary 1.5 [42]). We show that this claim is contradicted if there exists a polynomial-time algorithm that obtains an approximate solution, $\mathbf{x}$, to budget allocation satisfying

$$\min_{j\in 1,\dots,m} f_j(\mathbf{x}) \geq (1-\delta)\left(\min_{j\in 1,\dots,m} f_j(\mathbf{x}^*) - \gamma\right) \tag{B.3}$$

$$c(\mathbf{x}) \leq \eta B, \tag{B.4}$$

where $\mathbf{x}^*$ is an optimal solution and $\eta \leq (1-\alpha)(1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right)$ for a fixed $\alpha > 0$.

Assume such an algorithm exists. To satisfy Equation B.3, $\mathbf{x}$ must achieve a robust influence of $\min_{j\in 1,\dots,m} f_j(\mathbf{x}) = 1$, meaning that it is an optimal solution and therefore corresponds to a hitting set.

We can simplify the cost guarantee (Equation B.4) as follows: First, for this instance of robust budget alllocation, $\xi = \min_{f\in\Sigma} f(\mathbf{r}) = 1$ in the definition of $d_{\max}$ and $\beta_{\min}$. $d_{\max}$ is upper bounded by m, which occurs in the case that a single element constitutes a hitting set, where one node is activated in each of the m graph instances. Since $\xi = 1$, $H^{(\xi)}(\chi_s) = m$. The minimum marginal gain in $H^{(\xi)}$ occurs in the case that selecting an additional node (i.e. adding a unit vector) activates a target node in only of the graph instances, making the lower bound of $\beta_{\min}$ equal to 1. Thus, Equation B.4 can be written

$$(1-\alpha)(1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right) \leq (1-\alpha)(1+3\epsilon)\left(1+\log(m)\right).$$

We write

$$(1-\alpha)(1+\gamma)(1+\log(m)) = (1-(\alpha\gamma+\alpha-\gamma))(1+\log(m)).$$

For every $\alpha' < (\alpha\gamma+\alpha-\gamma)$, we can find an N such that for $m \geq N$,

$$(1-(\alpha\gamma+\alpha-\gamma))(1+\log(m)) \leq (1-\alpha')\log(m),$$

which sufficies to reach a contradiction.

□

Appendix C

Proof of DRPC Guarantee

The proof of Theorem 4.8 follows directly from the results of [41]. We restrict ourselves to considering additive cost functions, which fixes the curvature to $\rho = 1$. Moreover, we allow nodal budget capacities to vary from node to node.

Assume that the budget allocation, $\mathbf{x}$, will be updated a total of L times over the course of the algorithm. Let the subscript i denote variable values after the i-th update to $\mathbf{x}$. Define

$$\begin{aligned}
\mu_0 &:= 0 \\
\mu_i &:= \frac{k_i c(\chi_{s_i})}{f(k_i \chi_{s_i} | \mathbf{x}_{i-1})} \\
\hat{\mu}_0 &:= 0 \\
\hat{\mu}_i &:= \theta_i^{-1},
\end{aligned}$$

where θ_i is the threshold value after the i-th update of $\mathbf{x}$. Let $\{\mathbf{x}\}$ be the multiset where each $s \in S$ is contained $\mathbf{x}(s)$ times. We define the charge on element $s \in \{\mathbf{x}\}$ after update $1 \leq i \leq L$ to be

$$T_i(s, \mathbf{x}, f) := \mu_i (f(\chi_x | \mathbf{x}_{i-1}) - f(\chi_x | \mathbf{x}_i)).$$

The total charge on an allocation $\mathbf{x}$ is

$$\begin{aligned}
T(\mathbf{x}, f) &= \sum_{s \in \{\mathbf{x}\}} \sum_{i=1}^{L} T_i(s, \mathbf{x}, f) \\
&= \sum_{s \in \{\mathbf{x}\}} \sum_{i=1}^{L} \mu_i (f(\chi_s | \mathbf{x}_{i-1}) - f(\chi_s | \mathbf{x}_i)).
\end{aligned}$$

The proof of Theorem 4.8 relies on the following lemma and claims. We refer the reader to Soma and Yoshida, 2015 [41] for the proofs.

Lemma 6. (Soma and Yoshida, 2015 [41], Lemma 2.2) For a monotone DR-submodular function f and arbitrary $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_+^{|S|}$,

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \sum_{s \in \{\mathbf{x}\}} f(\chi_s | \mathbf{y}). \tag{C.1}$$

Claim 7. (Soma and Yoshida, 2015 [41], Claim 5.1) For any update $1 \leq i \leq L$,

$$\frac{f(k_i \chi_{s_i} | \mathbf{x}_{i-1})}{k_i} \geq \theta \tag{C.2}$$

and

$$f(\chi_{s_i} | \mathbf{x}_{i-1}) \leq \frac{\theta}{1-\epsilon}. \tag{C.3}$$

Combining these inequalities to eliminate θ, we obtain

$$\frac{k_i}{f(k_i \chi_{s_i} | \mathbf{x}_{i-1})} \leq \frac{1}{1-\epsilon} \leq \frac{1}{f(\chi_s | \mathbf{x}_{i-1})}.$$

Claim 8. (Soma and Yoshida, 2015 [41], Claim 5.3) For each $s \in \{\mathbf{x}^*\}$, the total charge on s is bounded by

$$\sum_{i=1}^{L} \mu_i (f(\chi_s | \mathbf{x}_{i-1}) - f(\chi_x | \mathbf{x}_i)) \leq \left(\frac{1}{1-\epsilon}\right) \left(1 + \log\left(\frac{d}{\beta}\right)\right).$$

Claim 9. (Soma and Yoshida, 2015 [41], Claim 5.2) The cost of allocation $\mathbf{x}$ is upper bounded by

$$c(\mathbf{x}) \leq \left(\frac{1}{1-\epsilon}\right) T(\mathbf{x}, f).$$

C.1 Proof of Theorem 4

Proof. To obtain the cost guarantee, we combine the previous claims to obtain the guarantee on the cost. (Note: The last inequality only holds for small ϵ, $0 < \epsilon \leq 0.232$.)

$$\begin{aligned}
c(\mathbf{x}) &\leq \left(\frac{1}{1-\epsilon}\right) T(\mathbf{x}, f) && \text{(Claim 9)} \\
&= \left(\frac{1}{1-\epsilon}\right) \sum_{s \in \{\mathbf{x}^*\}} \sum_{i=1}^{L} \mu_i (f(\chi_x | \mathbf{x}_{i-1}) - f(\chi_x | \mathbf{x}_i)) \\
&\leq \left(\frac{1}{1-\epsilon}\right) \sum_{s \in \mathbf{x}^*} \left(\frac{1}{1-\epsilon}\right) \left(1 + \log\left(\frac{d}{\beta}\right)\right) && \text{(Claim 8)} \\
&= \left(\frac{1}{1-\epsilon}\right)^2 \left(1 + \log\left(\frac{d}{\beta}\right)\right) c(\mathbf{x}^*) \\
&\leq (1+3\epsilon) \left(1 + \log\left(\frac{d}{\beta}\right)\right) c(\mathbf{x}^*)
\end{aligned}$$

To show that the objective function constraint is met, suppose we modify DRPC such that the outer loop terminated only when $f(\mathbf{x}) \leq \xi$ rather than when θ is sufficiently small. Let $\hat{\mathbf{x}}$ and $\mathbf{x}$ be the outputs of the modified DRPC and standard DRPC, respectively. We use Lemma 6 to bound the difference between $f(\hat{\mathbf{x}})$ and $f(\mathbf{x})$.

$$\begin{aligned}
f(\hat{\mathbf{x}}) - f(\mathbf{x}) &\leq \sum_{s \in \{\hat{\mathbf{x}}\}} f(\chi_s | \mathbf{x}) \\
&< \sum_{s \in \{\hat{\mathbf{x}}\}} \frac{\delta d}{n r_{\max}} \\
&= \frac{\delta d \, |\{\hat{\mathbf{x}}\}|}{n r_{\max}} \\
&\leq \delta d \\
&\leq \delta \xi,
\end{aligned}$$

where the second to last inequality holds because $|\{\hat{\mathbf{x}}\}| \leq n r_{\max}$. Rearranging the above expression, we obtain our objective constraint

$$f(\mathbf{x}) \geq f(\hat{\mathbf{x}}) - \delta\xi = (1-\delta)\xi.$$

□

Appendix D

Robust Ratio Formulation

D.1 Algorithm guarantee

The SaturateDR algorithm can also be applied to the robust ratio variation of the robust budget allocation problem, where the only change required is that the upper bound on ξ must be initialized to $\xi_{\max} \leftarrow 1$. The theoretical analysis remains nearly identical to that of Section 4.4 and is briefly outlined below. In this setting, SaturateDR achieves the following bi-criteria guarantee:

Theorem 10. *Given a fixed budget B and choosing an update size $\delta < \gamma/(2\min_{f\in\Sigma} f(\mathbf{r}))$,* SaturateDR *finds an approximate solution $\hat{\mathbf{x}}$ satisfying*

$$\rho(\hat{\mathbf{x}}) \geq (1-1/e)(1-\delta))(\rho(\mathbf{x}^*) - \gamma) \tag{D.1}$$

$$c(\hat{\mathbf{x}}) \leq (1+3\epsilon)\left(1+\log\left(\frac{d_{\max}}{\beta_{\min}}\right)\right)B, \tag{D.2}$$

where $\mathbf{x}^*$ is the optimal solution to the robust ratio version of robust budget allocation (Problem 3.3), $0 < \epsilon, \gamma < 1$ are approximation parameters, $d_{\max} = \max_{s\in S} H_\rho^{(\xi)}(\chi_s)$, and $\beta_{\min} = \min\{H_\rho^{(\xi)}(\chi_s|\mathbf{x}) : s \in S, \mathbf{x} \in \mathbb{Z}_+^S, H_\rho^{(\xi)}(\chi_s|\mathbf{x}) > 0\}$, where $\xi = 1$.

Proof. One key difference in using the robust ratio is that the definition requires $\mathbf{x}_f^*$, the optimal allocation for influence function f. Recall that this is NP-hard to find so we instead consider the "greedy robust ratio":

$$\rho^g(\mathbf{x}) = \min_{f\in\Sigma} \frac{f(\mathbf{x})}{f(\mathbf{x}_f^g)}, \tag{D.3}$$

where $f(\mathbf{x}_f^g)$ is a greedy approximation of $f(\mathbf{x}_f^*)$. For any allocation, $\mathbf{x}$,

$$(1-1/e)\rho^g(\mathbf{x}) \leq \rho(\mathbf{x}) \leq \rho^g(\mathbf{x}), \tag{D.4}$$

is satisfied. To be able to simultaneously optimize over all influence functions, for a given constant ξ, we define

$$H_\rho^{(\xi)}(\mathbf{x}) := \sum_{f \in \Sigma} \min\left(\xi, \frac{f(\mathbf{x})}{f(\mathbf{x}_f^g)}\right), \tag{D.5}$$

which is monotone DR-submodular. We observe that $\rho^g(\mathbf{x}) \geq \xi$ if and only if $H_\rho^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi$. Analogous to the problem reformulation in Section 4.1, we re-write Problem 3.3 as

$$\begin{aligned} \max_{\xi, \mathbf{x}} \xi \qquad \text{s.t. } & H_\rho^{(\xi)}(\mathbf{x}) \geq |\Sigma|\,\xi \\ & c(\mathbf{x}) \leq \eta B \\ & \mathbf{x} \leq \mathbf{r}. \end{aligned} \tag{D.6}$$

The remainder of the proof is nearly identical to the proof of Theorem 3. Because $\xi_{\max}$ and $\xi_{\min}$ are updated in the same manner as in Section 4.4, we know that the binary search for the optimal ξ will terminate in $\mathcal{O}(1/\gamma)$ iterations.

To prove that the guarantee on $\rho(\hat{\mathbf{x}})$ (Equation D.1) is satisfied, we similarly note that the resulting allocation, $\hat{\mathbf{x}}$, from SaturateDR is the allocation outputted by DRPC using an optimal ξ^*. The guarantee from DRPC implies that $\rho^g(\hat{\mathbf{x}}) \geq (1-\delta)\xi^*$. Additionally, we have that $\xi^* \geq \xi_{\max} - \gamma$ must hold during the last iteration of SaturateDR and that $\xi_{\max} \geq \rho(\mathbf{x}^*)$ due to the definition of $\xi_{\max}$. Combining these inequalities with Equation D.4 gives

$$\begin{aligned} \rho(\hat{\mathbf{x}}) &\geq (1 - 1/e)\rho^g(\hat{\mathbf{x}}) \\ &\geq (1 - 1/e)(1-\delta)\xi^* \\ &\geq (1 - 1/e)(1-\delta)(\xi_{\max} - \gamma) \\ &\geq (1 - 1/e)(1-\delta)(\rho(\mathbf{x}^*) - \gamma), \end{aligned}$$

finishing the proof of Theorem 10. □

Bibliography

[1] Noga Alon, Iftah Gamzu, and Moshe Tennenholtz. "Optimizing budget allocation among channels and influencers". In: *Proceedings of the 21st international conference on World Wide Web*. ACM. 2012, pp. 381–388.

[2] Pedro Domingos and Matt Richardson. "Mining the Network Value of Customers". In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '01. San Francisco, California: ACM, 2001, pp. 57–66. ISBN: 1-58113-391-X. DOI: 10.1145/502512.502525. URL: http://doi.acm.org/10.1145/502512.502525.

[3] Matthew Richardson and Pedro Domingos. "Mining Knowledge-sharing Sites for Viral Marketing". In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada: ACM, 2002, pp. 61–70. ISBN: 1-58113-567-X. DOI: 10.1145/775047.775057. URL: http://doi.acm.org/10.1145/775047.775057.

[4] David Kempe, Jon Kleinberg, and Éva Tardos. "Maximizing the Spread of Influence Through a Social Network". In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '03. Washington, D.C.: ACM, 2003, pp. 137–146. ISBN: 1-58113-737-0. DOI: 10.1145/956750.956769. URL: http://doi.acm.org/10.1145/956750.956769.

[5] Tasuku Soma et al. "Optimal Budget Allocation: Theoretical Guarantee and Efficient Algorithm". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 1. Bejing, China: PMLR, June 2014, pp. 351–359. URL: http://proceedings.mlr.press/v32/soma14.html.

[6] Kazumi Saito, Ryohei Nakano, and Masahiro Kimura. "Prediction of Information Diffusion Probabilities for Independent Cascade Model". In: *Proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III*. KES '08. Zagreb, Croatia: Springer-Verlag, 2008, pp. 67–75. ISBN: 978-3-540-85566-8. DOI: 10.1007/978-3-540-85567-5_9. URL: http://dx.doi.org/10.1007/978-3-540-85567-5_9.

[7] Praneeth Netrapalli and Sujay Sanghavi. "Learning the Graph of Epidemic Cascades". In: *SIGMETRICS Perform. Eval. Rev.* 40.1 (June 2012), pp. 211–222. ISSN: 0163-5999. DOI: 10.1145/2318857.2254783. URL: http://doi.acm.org/10.1145/2318857.2254783.

[8] Manuel Gomez-Rodriguez et al. "Estimating Diffusion Networks: Recovery Conditions, Sample Complexity and Soft-thresholding Algorithm". In: *Journal of Machine Learning Research* 17.90 (2016), pp. 1–29. URL: http://jmlr.org/papers/v17/14-430.html.

[9] Lily Hu et al. "Activating the "Breakfast Club": Modeling Influence Spread in Natural-World Social Networks". In: *CoRR* abs/1710.00364 (2017). arXiv: 1710.00364. URL: http://arxiv.org/abs/1710.00364.

[10] Noa Avigdor-Elgrabli, Gideon Blocq, and Iftah Gamzu. "Offline and Online Models of Budget Allocation for Maximizing Influence Spread". In: *CoRR* abs/1508.01059 (2015). arXiv: 1508.01059. URL: http://arxiv.org/abs/1508.01059.

[11] Daisuke Hatano et al. "Lagrangian Decomposition Algorithm for Allocating Marketing Channels". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI'15. Austin, Texas: AAAI Press, 2015, pp. 1144–1150. ISBN: 0-262-51129-0. URL: http://dl.acm.org/citation.cfm?id=2887007.2887166.

[12] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. "An analysis of approximations for maximizing submodular set functions—I". In: *Mathematical Programming* 14.1 (Dec. 1978), pp. 265–294. ISSN: 1436-4646. DOI: 10.1007/BF01588971. URL: https://doi.org/10.1007/BF01588971.

[13] Maxim Sviridenko. "A Note on Maximizing a Submodular Set Function Subject to a Knapsack Constraint". In: *Oper. Res. Lett.* 32.1 (Jan. 2004), pp. 41–43. ISSN: 0167-6377. DOI: 10.1016/S0167-6377(03)00062-2. URL: http://dx.doi.org/10.1016/S0167-6377(03)00062-2.

[14] Michel Minoux. "Accelerated greedy algorithms for maximizing submodular set functions". In: *Optimization Techniques*. Ed. by J. Stoer. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, pp. 234–243. ISBN: 978-3-540-35890-9.

[15] Jure Leskovec et al. "Cost-effective Outbreak Detection in Networks". In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '07. San Jose, California, USA: ACM, 2007, pp. 420–429. ISBN: 978-1-59593-609-7. DOI: 10.1145/1281192.1281239. URL: http://doi.acm.org/10.1145/1281192.1281239.

[16] Wei Chen, Yifei Yuan, and Li Zhang. "Scalable Influence Maximization in Social Networks Under the Linear Threshold Model". In: *Proceedings of the 2010 IEEE International Conference on Data Mining*. ICDM '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 88–97. ISBN: 978-0-7695-4256-0. DOI: 10.1109/ICDM.2010.118. URL: http://dx.doi.org/10.1109/ICDM.2010.118.

[17] Masahiro Kimura and Kazumi Saito. "Tractable Models for Information Diffusion in Social Networks". In: *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*. ECMLPKDD'06. Berlin, Germany: Springer-Verlag, 2006, pp. 259–271. ISBN: 3-540-45374-1, 978-3-540-45374-1. DOI: 10.1007/11871637_27. URL: https://doi.org/10.1007/11871637_27.

[18] Wei Chen, Yajun Wang, and Siyu Yang. "Efficient Influence Maximization in Social Networks". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. Paris, France: ACM, 2009, pp. 199–208. ISBN: 978-1-60558-495-9. DOI: 10.1145/1557019.1557047. URL: http://doi.acm.org/10.1145/1557019.1557047.

[19] Amit Goyal, Francesco Bonchi, and Laks V. S. Lakshmanan. "A Data-based Approach to Social Influence Maximization". In: *Proc. VLDB Endow.* 5.1 (Sept. 2011), pp. 73–84. ISSN: 2150-8097. DOI: 10.14778/2047485.2047492. URL: http://dx.doi.org/10.14778/2047485.2047492.

[20] Baharan Mirzasoleiman et al. "Lazier Than Lazy Greedy". In: *CoRR* abs/1409.7938 (2014). arXiv: 1409.7938. URL: http://arxiv.org/abs/1409.7938.

[21] Jan Vondrak. "Optimal Approximation for the Submodular Welfare Problem in the Value Oracle Model". In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. STOC '08. Victoria, British Columbia, Canada: ACM, 2008, pp. 67–74. ISBN: 978-1-60558-047-0. DOI: 10.1145/1374376.1374389. URL: http://doi.acm.org/10.1145/1374376.1374389.

[22] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. "Dependent randomized rounding for matroid polytopes and applications". In: *arXiv preprint arXiv:0909.4348* (2009).

[23] Chandra Chekuri and Jan Vondrák. "Randomized Pipage Rounding for Matroid Polytopes and Applications". In: *CoRR* abs/0909.4348 (2009). arXiv: 0909.4348. URL: http://arxiv.org/abs/0909.4348.

[24] Gruia Calinescu et al. "Maximizing a Monotone Submodular Function Subject to a Matroid Constraint". In: *SIAM J. Comput.* 40.6 (Dec. 2011), pp. 1740–1766. ISSN: 0097-5397. DOI: 10.1137/080733991. URL: http://dx.doi.org/10.1137/080733991.

[25] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. "A Unified Continuous Greedy Algorithm for Submodular Maximization". In: *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*. FOCS '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 570–579. ISBN: 978-0-7695-4571-4. DOI: 10.1109/FOCS.2011.46. URL: http://dx.doi.org/10.1109/FOCS.2011.46.

[26] Ashwinkumar Badanidiyuru and Jan Vondrák. "Fast Algorithms for Maximizing Submodular Functions". In: *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '14. Portland, Oregon: Society for Industrial and Applied Mathematics, 2014, pp. 1497–1514. ISBN: 978-1-611973-38-9. URL: http://dl.acm.org/citation.cfm?id=2634074.2634184.

[27] Chandra Chekuri, T.S. Jayram, and Jan Vondrak. "On Multiplicative Weight Updates for Concave and Submodular Function Maximization". In: *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*. ITCS '15. Rehovot, Israel: ACM, 2015, pp. 201–210. ISBN: 978-1-4503-3333-7. DOI: 10.1145/2688073.2688086. URL: http://doi.acm.org/10.1145/2688073.2688086.

[28] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. "Inferring Networks of Diffusion and Influence". In: *ACM Trans. Knowl. Discov. Data* 5.4 (Feb. 2012), 21:1–21:37. ISSN: 1556-4681. DOI: 10.1145/2086737.2086741. URL: http://doi.acm.org/10.1145/2086737.2086741.

[29] Nan Du et al. "Scalable Influence Estimation in Continuous-Time Diffusion Networks". In: *Advances in Neural Information Processing Systems 26*. Ed. by C. J. C. Burges et al. Curran Associates, Inc., 2013, pp. 3147–3155. URL: http://papers.nips.cc/paper/4857-scalable-influence-estimation-in-continuous-time-diffusion-networks.pdf.

[30] Nan Du et al. "Influence Function Learning in Information Diffusion Networks". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. ICML'14. Beijing, China: JMLR.org, 2014, pp. II-2016–II-2024. URL: http://dl.acm.org/citation.cfm?id=3044805.3045117.

[31] Xinran He and David Kempe. "Stability of Influence Maximization". In: *CoRR* abs/1501.04579 (2015). arXiv: 1501.04579. URL: http://arxiv.org/abs/1501.04579.

[32] Wei Chen et al. "Robust Influence Maximization". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, 2016, pp. 795–804. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939745. URL: http://doi.acm.org/10.1145/2939672.2939745.

[33] Dimitris Kalimeris, Gal Kaplun, and Yaron Singer. "Robust Influence Maximization for Hyperparametric Models". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, Sept. 2019, pp. 3192–3200. URL: http://proceedings.mlr.press/v97/kalimeris19a.html.

[34] Andreas Krause et al. "Robust submodular observation selection". In: *Journal of Machine Learning Research* 9.Dec (2008), pp. 2761–2801.

[35] Xinran He and David Kempe. "Robust Influence Maximization". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, 2016, pp. 885–894. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939760. URL: http://doi.acm.org/10.1145/2939672.2939760.

[36] Thomas Powers et al. "Constrained Robust Submodular Sensor Selection with Applications to Multistatic Sonar Arrays". In: *19th International Conference on Information Fusion*. Heidelberg, Germany: IEEE, July 2016.

[37] Thomas Powers et al. "Constrained Robust Submodular Optimization". In: 2016.

[38] Nima Anari et al. "Structured Robust Submodular Maximization: Offline and Online Algorithms". In: *Proceedings of Machine Learning Research*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 16–18 Apr 2019, pp. 3128–3137. URL: http://proceedings.mlr.press/v89/anari19a.html.

[39] Rajan Udwani. "Multi-objective Maximization of Monotone Submodular Functions with Cardinality Constraint". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., 2018, pp. 9493–9504. URL: http://papers.nips.cc/paper/8159-multi-objective-maximization-of-monotone-submodular-functions-with-cardinality-constraint.pdf.

[40] Matthew Staib and Stefanie Jegelka. "Robust Budget Allocation via Continuous Submodular Functions". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 3230–3240. URL: http://proceedings.mlr.press/v70/staib17a.html.

[41] Tasuku Soma and Yuichi Yoshida. "A generalization of submodular cover via the diminishing return property on the integer lattice". In: *Advances in Neural Information Processing Systems*. 2015, pp. 847–855.

[42] Irit Dinur and David Steurer. "Analytical Approach to Parallel Repetition". In: *CoRR* abs/1305.1979 (2013). arXiv: 1305.1979. URL: http://arxiv.org/abs/1305.1979.