Minimum Robust Multi-Submodular Cover for Fairness

Lan N. Nguyen My T. Thai
Department of Computer and Information Science and Engineering
University of Florida, Gainesville, Florida 32611
lan.nguyen@ufl.edu mythai@cise.ufl.edu

Abstract

In this paper, we study a novel problem, **Minimum Robust Multi-Submodular Cover for Fairness** (MINRF), as follows: given a ground set V; m monotone submodular functions $f_1, ..., f_m$; m thresholds $T_1, ..., T_m$ and a non-negative integer r, MINRF asks for the smallest set S such that for all $i \in [m]$, $\min_{|X| \leq r} f_i(S \setminus X) \geq T_i$. We prove that MINRF is inapproximable within $(1 - \epsilon) \ln m$; and no algorithm, taking fewer than exponential number of queries in term of r, is able to output a feasible set to MINRF with high certainty. Three bicriteria approximation algorithms with performance guarantees are proposed: one for $r = 0$, one for $r = 1$, and one for general r. We further investigate our algorithms' performance in two applications of MINRF, Information Propagation for Multiple Groups and Movie Recommendation for Multiple Users. Our algorithms have shown to outperform baseline heuristics in both solution quality and the number of queries in most cases.

Introduction

In a minimum submodular cover, given a ground set V, a monotone submodular set function $f : 2^V \to \mathbb{R}$ and a number T, the problem asks for a set $S \subseteq V$ of minimum size such that $f(S) \geq T$. This problem was studied extensively in the literature because of its wide-range applications, e.g. data summarization (Mirzasoleiman et al. 2015; Mirzasoleiman, Zadimoghaddam, and Karbasi 2016), active set selection (Norouzi-Fard et al. 2016), recommendation systems (Guillory and Bilmes 2011), information propagation in social networks (Kuhnle et al. 2017), and network resilience assessment (Nguyen and Thai 2019; Dinh and Thai 2014).

However, a single objective function f may not well model several practical applications where achieving multiple goals is required, especially when group fairness is considered. Let us consider the following two representative applications.

Information Propagation in Social Network for Multiple Groups. Social networks are cost-effective tools for information spreading by selecting a set of highly influential people (called seed set) that, through the word-of-mouth effects, the information will be reached to a large number of population (Kuhnle et al. 2017; Nguyen, Zhou, and Thai 2019; Zhang et al. 2014; Nguyen, Thai, and Dinh 2016). For many applications (e.g. broadening participants in STEM), it is important to ensure the diversity and fairness among different ethnics and genders. Therefore, those applications aim to find a minimum seed set such that the information can reach to each group in a fair manner.

Items Recommendation for Multiple Users. Recommendation systems aim to make a good recommendation, e.g. a set of items, which can match users' preferences. In many situations, an item can be served for multiple users, e.g. a family. In this problem, a user's utility level to a set of items is modelled under a monotone submodular function. The objective, therefore, is to find the smallest set of items, from which we can design a recommendation for all users in a way that all reach a certain utility level.

Additionally, these problems require robustness in the solution set, in the sense that the solution satisfies all the constraints even if some elements were removed. Those removal can be from various reasons. For instance, in information propagation, a subset of users may decide not to spread the information (Bogunovic et al. 2017). Or in recommendation systems, due to the uncertainty of underlying data, information of some items may not be accurate (Orlin, Schulz, and Udwani 2018).

Achieving a reasonable prior distribution on the removed elements may not be practical in many situations. Or even when the distribution is known, it is critical to obtain a robust solution with a high level of certainty, in a way that all goals are still achieved under the worst-case removal. Motivated by that observation, in this work, we study a novel problem, **Minimum Robust Multi-Submodular Cover for Fairness** (MINRF), defined as follows.

Definition 1. (MINRF) Given a finite set V; m monotone submodular functions $f_1, ... f_m$ where $f_i : 2^V \to \mathbb{R}^{\geq}$; m non-negative numbers $T_1, ..., T_m$, and a non-negative integer r, find a set $S \subseteq V$ of minimum size such that $\forall\ i \in [m]$, $\min_{|X| \leq r} f_i(S \setminus X) \geq T_i$.

MINRF's objective can also be understood as finding S of minimum size such that for all $X \subseteq V$ that $|X| \leq r$ and $i \in [m]$, $f_i(S \setminus X) \geq T_i$. Beside two applications as stated early, MINRF can also be applied in many other applications, such as **Sensor Placement** (Orlin, Schulz, and Udwani 2018; Ohsaka and Yoshida 2015), which guarantees each measurement (e.g. temperature, humidity) reaches a certain information gain while being robust against sensors' failure;

or **Feature Selection** (Qian et al. 2017; Orlin, Schulz, and Udwani 2018), which aims for a smallest set of features that can retain information at a certain level while guaranteeing the set is not dependent on a few features.

To solve MINRF, one direction is to list all constraints in a form of $f_i(. \setminus X) \geq T_i \; \forall |X| = r$ and $i \in [m]$ and find a smallest set that satisfies all those constraints. However, with large V, the amount of set X of size r is $\binom{n}{r}$, making it impractical to enumerate all possible removed sets. Furthermore, we show that an algorithm, which is able to output a feasible solution to MINRF in general, is very expensive, requiring at least exponential number of queries in term of r. Even when $r = 0$, there exists no polynomial algorithm that can approximate MINRF within a factor of $(1-\epsilon)\ln m$ unless $P = NP$. Thus solving MINRF remains open and to our knowledge, we are the first one studying the problem.

Contribution. Beside introducing MINRF and investigating the problem's hardness using complexity theory, we propose a bicriteria approximation algorithm, namely ALGR, to solve MINRF. ALGR's performance guarantee is tight to MINRF's inapproximability and required query complexity. To be specific, ALGR is polynomial with fixed r and obtain S of size $O(\ln \frac{\max(m,n)}{\alpha})$ factor to the optimal solution where $\alpha \in (0, 1]$. S guarantees that for all $X \subset V$ that $|X| \leq r$ and $i \in [m]$, $f_i(S \setminus X) \geq (1-\alpha)T_i$. In a special case of $r = 1$, we propose ALG1 which can run faster than ALGR.

Both ALGR and ALG1 work in a manner that they frequently call an algorithm solving MINRF with $r = 0$ as a subroutine. Although MINRF with $r = 0$ has been studied in the literature, a new aspect of the problem requires us to propose new solutions to MINRF where $r = 0$. In particular, we propose Random Greedy (RANDGR) and re-investigate two existing algorithms, GREEDY and THRESGR, whose performance has been analyzed where f_is receive values in $\mathbb{Z}$, in order to adapt them to $\mathbb{R}$ domain. In comparison to GREEDY and THRESGR, RANDGR does not unite submodular functions into a single function; and introduces randomness to reduce queries to f_is. RANDGR takes much fewer queries than GREEDY and THRESGR as shown in our experiments.

Further, we investigate our algorithms' performance on two applications of MINRF: Information Propagation for Multiple Groups and Movie Recommendation for Multiple Users. The experimental results show our algorithms outperformed some intuitive heuristics methods in both quality of solutions and the number of queries.

Preliminaries

Related Work

To our knowledge, this work provides the first solutions to MINRF for a general r. In this part, we pay attention to recent studies on minimum multi-submodular cover (MINRF when $r = 0$), and robust submodular optimization.

With minimum submodular cover ($m = 1, r = 0$), Goyal et al. (2013) showed that the classical greedy algorithm is able to obtain a bi-criteria ratio of $O(\ln \alpha^{-1})$. If we run m instances of greedy, each with a constraint $f_i(.) \geq T_i$, get m output S_i and returns $\cup_{i \in [m]} S_i$, we can get the ratio of $O(m \ln \alpha^{-1})$ for MINRF when $r = 0$. In this paper, we aim for algorithms with better ratios.

Krause et al. (2008) was the first one proposing a problem of minimum multi-submodular cover; and the problem was then further studied by Mirzasoleiman, Zadimoghaddam, and Karbasi (2016); Iyer and Bilmes (2013). In general, their solution made a reduction from multiple submodular objectives to a single instance of a submodular cover problem by defining $F(.) = \sum_{i \in [m]} \min(f_i(.), T)$ (all thresholds are the same); and find S of minimum size such that $F(S) = mT$. Two algorithms were proposed, GREEDY (Krause et al. 2008; Iyer and Bilmes 2013) and THRESGR (Mirzasoleiman, Zadimoghaddam, and Karbasi 2016). Their performance analysis requires $\{f_i\}_{i \in [m]}$ to receive values in $\mathbb{Z}$ to obtain ratio of $O(\ln \max_{e \in V} F(\{e\}))$.

However, requiring $\{f_i\}_{i \in [m]}$ to receive values in $\mathbb{Z}$ is not practical in many applications. In our work, we re-investigate GREEDY and THRESGR's performance without such requirement. Also, our RANDGR algorithm differs from such methods in which RANDGR does not unite objectives into a single function. Furthermore, RANDGR adds randomness to reduce the query complexity while still obtaining an asymptotically equal performance guarantee to that of GREEDY.

With robust submodular optimization, the concept of finding set that is robust to the removal of r elements was first proposed by Orlin, Schulz, and Udwani (2018). However, their problem is a maximization, namely Robust Submodular Maximization (RSM), defined as follows: Given a ground set V, a monotone submodular function f, nonnegative integers k and r, find S s.t $|S| \leq k$ that maximizes $\min_{Z \subseteq A, |Z| \leq r} f(A \setminus Z)$. This problem was later studied further by Bogunovic et al. (2017); Mitrovic et al. (2017); Staib, Wilder, and Jegelka (2019); Anari et al. (2019). RSM and MINRF both focus on the worst-case scenario, where the removal of r elements has the greatest impact on the returned solution. Other than that, the two problems are basically different and we are unable to adapt existing algorithms for RSM to solve MINRF with performance guarantees. The key bottleneck preventing us to adapt those algorithms is how to guarantee that a returned solution is robust and satisfied submodular constraints.

Definitions & Complexity

In this part, we present definitions and theories that would be used frequently in our analysis; and analyze complexity of solving MINRF. Due to page limit, detailed proofs of lemmas and theorems of this part are provided in Appendix.

Definition 2. Given an instance of MINRF, including $V, \{f_i\}_{i \in [m]}, \{T_i\}_{i \in [m]}$, a set $A \subseteq V$ is (t, α)**-robust** iff for all $i \in [m]$, $\min_{|X| \leq t} f_i(A \setminus X) \geq (1-\alpha)T_i$.

Speaking in another way, MINRF asks us to find a minimum $(r, 0)$-robust set.

Without loss of generality, in our algorithm, we change $f_i(.) := \min(f_i(.)/T_i, 1)$. It is trivial that f_i is still monotone submodular; and MINRF's objective now is to find S that $\min_{|X| \leq r} f_i(S \setminus X) \geq 1 \; \forall \, i \in [m]$.

If there exists a $(r, 0)$-robust set, denote S^* as an optimal solution; and $OPT(U, t)$ as a size of the minimum $(t, 0)$-

robust set that is a subset of U if there is any. So $|S^*| = OPT(V, r)$. We have the following key lemma:

Lemma 1. For all $X_1, X_2 \subseteq V$ that $|X_1| = r_1$, $|X_2| = r_2$ and $r_1 + r_2 \leq r$

$$\begin{aligned} OPT(V, r) &\geq OPT(V \setminus X_1, r - r_1) \\ &\geq OPT\big(V \setminus (X_1 \cup X_2), r - r_1 - r_2\big) \\ &\geq OPT(V, 0) \end{aligned}$$

Lemma 1 is very critical and will be used frequently to obtain performance guarantees of our algorithms.

Given a MINRF instance and $\alpha \in [0, 1]$, we aims to devise algorithms that guarantee:

- If there exists (r, α)-robust sets in the MINRF instance, the returned solution is (r, α)-robust with size at most some factor to $OPT(V, r)$.
- Otherwise, the algorithms notify no $(r, 0)$-robust set exists.

We first study the hardness of devising such an algorithm to solve MINRF. First, we show that: even the sub-task of outputting a $(r, 0)$-robust set if there is any, is already very expensive. That is stated in the following theorem.

Theorem 1. There exists no algorithm, taking fewer than exponential number of queries in term of r, is able to verify existence of a $(r, 0)$-robust set to MINRF.

Theorem 1 is proven by taking one instance of MINRF, in which the removal of any subset $X \subseteq V$ of a same size shows a similar behavior on the submodular objectives except for only one unique subset R of size r. The thresholds $\{T_i\}_i$ are set so that: if there exists a $(r, 0)$-robust set then V is the only $(r, 0)$-robust set and R is the only set that would make $V \setminus R$ violate the constraints. Thus any algorithm, taking fewer than $O(\binom{|V|}{r})$ queries is unable to verify whether V is $(r, 0)$-robust. The full description of the MINRF instance is provided in Appendix.

Furthermore, even there exists $(r, 0)$-robust sets, devising approximation algorithms for MINRF is NP-hard. We have the following theorem.

Theorem 2. There exists no polynomial algorithm that can approximate MINRF, even with $r = 0$, within a factor of $(1 - \epsilon) \ln m$ given $\epsilon > 0$ unless $P = NP$.

Algorithms When $r = 0$

We first study MINRF with $r = 0$ since complexity and solution quality of algorithms for MINRF with $r = 0$ play critical roles on the performance of ALG1 and ALGR. Although MINRF with $r = 0$ has been studied in the literature, these results cannot applied directly. The key barrier is that the initial solution set may not be empty.

In this part, we propose RANDGR, a randomized algorithm with bicriteria approximation ratio of $O(\ln \frac{m}{\alpha})$. Also, we re-investigate performance guarantees of GREEDY and THRESGR, extending from their performance when f_is receive values in $\mathbb{Z}$.

With RANDGR, checking if there exists feasible solutions with $r = 0$ is quite trivial. RANDGR simply verifies whether

Algorithm 1 RANDGR

Input $V, S_0, \{f_i\}_{i \in [m]}$

if There exists $i \in [m]$ s.t. $f_i(V) < 1 - \alpha$ **then**
  Return no feasible solution
$t = 0$; $\mathcal{F}_0 = \{f_i\}_{i \in [m]}$
while $\mathcal{F}_t \neq \emptyset$ **do**
  F = randomly select $\lceil |\mathcal{F}_t|/2 \rceil$ constraints from $\mathcal{F}_t$
  $e_t = \text{argmax}_{e \in V \setminus S_t} \sum_{f_i \in F} \Delta_e f_i(S_t)$
  $S_{t+1} = S_t \cup \{e_t\}$; $\mathcal{F}_{t+1} = \mathcal{F}_t$; $t = t + 1$
  Remove all $f_i \in \mathcal{F}_t$ that $f_i(S_t) \geq 1 - \alpha$ out of $\mathcal{F}_t$

Return S_t

$f_i(V) \geq 1 - \alpha$ for all $i \in [m]$. If no, the algorithm notifies no feasible set exists and terminates.

If there exists feasible solutions, RANDGR works in rounds in order to find a $(0, \alpha)$-robust solution. For each round, a new random process is introduced as follows: the algorithm randomly selects half of functions f_is, each of which is still less than $1 - \alpha$; and greedily chooses an element that maximizes the sum of marginal gains of the selected functions. This random process helps RANDGR (1) reduce the number of queries to f_is by half at each round; and (2) establish a recursive relationship of obtained solutions at different rounds, which is critical for RANDGR to obtain its performance guarantee with high probability (w.h.p).

RANDGR's pseudocode is presented by Alg. 1. In Alg. 1, S_t represents an obtained solution at round t and $\mathcal{F}_t$ is a set of f_is that $f_j(S_t) \geq 1 - \alpha \; \forall \; f_j \notin \mathcal{F}_t$. Note that RANDGR starts with S_0 as an input; and as can be seen later, RANDGR is used as a subroutine function in case $r > 0$, in which S_0 may not be empty. Therefore, analyzing performance of RANDGR with $S_0 \neq \emptyset$ is necessary and challenging.

To obtain RANDGR's performance guarantee, we have the following lemma.

Lemma 2. At round t: $\mathrm{E}\Big[\sum_{f_i \in \mathcal{F}_{t+1}} \big(1 - f_i(S_{t+1})\big)\Big] \leq \big(1 - \frac{1}{2\,OPT(V,0)}\big) \sum_{f_i \in \mathcal{F}_t} \big(1 - f_i(S_t)\big)$

Lemma 2 establishes a recursive relationship between $\mathcal{F}_t$ and S_t at different rounds. This is a key to obtain RANDGR's approximation ratio. Assuming RANDGR stops after L rounds, $L = |S_L \setminus S_0|$. By using Markov inequality, we can bound L w.h.p to obtain RANDGR's performance guarantee as Theorem 3. Full proofs of Lemma 2 and Theorem 3 are presented in Appendix.

Theorem 3. Given an instance of MINRF with input $V, \{f_i\}_{i \in [m]}, S_0$ such that $\sum_{i \in [m]} f_i(S_0) \geq (1 - \eta)m$, and $r = 0$. If S is an output of RANDGR then w.h.p $|S \setminus S_0| \leq OPT(V, 0) O(\ln \frac{m\eta}{\alpha})$ and each f_i is queried at most $O(|V| OPT(V, 0) \ln \frac{m\eta}{\alpha})$ times.

We now investigate the performance of GREEDY and THRESGR. Their performance guarantees are stated by Theorem 4 (GREEDY) and 5 (THRESGR). Due to page limit, their detailed description and proofs are presented in Appendix.

Theorem 4. Given an instance of MINRF with input $V, \{f_i\}_{i\in[m]}, S_0$ such that $\sum_{i\in[m]} f_i(S_0) \geq (1-\eta)m$, and $r = 0$. If GREEDY terminates with a $(0, \alpha)$-robust solution S, then $|S \setminus S_0| \leq OPT(V, 0)O(\ln \frac{m\eta}{\alpha})$ and each f_i is queried at most $O(|V|OPT(V, 0) \ln \frac{m\eta}{\alpha})$ times.

Theorem 5. Given an instance of MINRF with input $V, \{f_i\}_{i\in[m]}, S_0$ such that $\sum_{i\in[m]} f_i(S_0) \geq (1-\eta)m$, and $r = 0$. If THRESGR terminates with a $(0, \alpha)$-robust solution S, then $|S \setminus S_0| \leq OPT(V, 0)O(\frac{1}{1-\gamma} \ln \frac{m\eta}{\alpha})$ where $\gamma \in (0, 1)$ is the algorithm's parameter; and each f_i is queried at most $O(\frac{n}{\gamma} \ln \frac{mn}{\alpha})$ times.

With $S_0 = \emptyset$, RANDGR, GREEDY and THRESGR (γ is close to 0) can obtain a ratio of $O(\ln \frac{m}{\alpha})$, which is tight to the inapproximability of MINRF when $r = 0$ (Theorem 2).

Algorithms When $r > 0$

In this section, we propose two algorithms to solve MINRF when $r > 0$: ALG1 for a special case of $r = 1$ and ALGR for general r. Both algorithms frequently call an algorithm to MINRF when $r = 0$ as a subroutine, which could be either RANDGR, GREEDY or THRESGR as discussed earlier. In short, we use ALG0 to refer to any of these three.

For simplicity, we ignore the step of notifying if there exists no (r, α) robust set in ALG1 and ALGR's description since it can trivially inferred from the outputs of ALG0. Without loss of generality, in our analysis, we assume there exists $(r, 0)$-robust sets.

Algorithm When $r = 1$ (ALG1)

In general, ALG1 is an iterative algorithm, which iteratively checks if there exists an element whose removal causes an obtained solution S to violate at least one constraint. If such an element (let's call it e) exists, ALG1 gathers all violated constraints to form a new MINRF instance with $r = 0$, $V \setminus \{e\}$ as an input ground set and $S \setminus \{e\}$ as an initial set. This is a key of ALG1 because by solving that new MINRF instance using ALG0, ALG1 guarantees the obtained solution is robust against e's removal; and the algorithm can significantly tighten an upper bound on the number of newly-added elements in order to obtain a tight approximation ratio.

ALG1's pseudocode is presented by Alg. 2. In Alg. 2, S_1 is a $(0, \alpha)$-robust set, found by using ALG0 with the original MINRF's input (line 1). S, returned by ALG1, is $(1, \alpha)$-robust because:

- For any $e \in S_1$ that violates the condition of **while** loop (line 2), ALG1 guarantees $f_i(S \setminus \{e\}) \geq 1 - \alpha$ (output of ALG0, line 4).
- For any $e \notin S_1$, as $S_1 \subseteq S \setminus \{e\}$, we have $f_i(S \setminus \{e\}) \geq f_i(S_1) \geq 1 - \alpha$ (output of ALG0, line 1).

Denote E as a set of $e \in S_1$ that violate the condition of **while** loop (line 2). For each $e \in E$, denote S^e as S right before e is considered by the **while** loop of line 2. Let $\rho_e = \sum_i \Delta_e f_i(S^e \setminus \{e\})/m$. To obtain ALG1's performance guarantee, we have the following lemma.

Lemma 3. $\sum_{e\in E} \rho_e \leq 1$

Algorithm 2 Algorithm when $r = 1$ (ALG1)

Input $V, \{f_i\}_{i\in[m]}$

1: $S = S_1 =$ ALG0 $(V, \emptyset, \{f_i\}_{i\in[m]})$
2: **while** $\exists e \in S_1$ that $\exists\, i \in [m]$, $f_i(S \setminus \{e\}) < 1 - \alpha$ **do**
3: $\quad \mathcal{F}' =$ set of all f_i that $f_i(S \setminus \{e\}) < 1 - \alpha$
4: $\quad S' =$ ALG0 $(V \setminus \{e\}, S \setminus \{e\}, \mathcal{F}')$
5: $\quad S = S \cup S'$

Return S

Proof. Let's sort elements in $S_1 = \{u_1, u_2, ...\}$ in the order of being added into S_1 by ALG1 (line 1). Let $S_1^e = \{u_1, ..., u_{e-1}\}$. Due to submodularity, $\sum_i \Delta_e f_i(S_1^e) \geq \sum_i \Delta_e f_i(S \setminus \{e\}) = \rho_e m$. Then:

$$\sum_{e\in E} \rho_e m \leq \sum_{e\in E} \sum_i \Delta_e f_i(S_1^e) \leq \sum_{e\in S_1} \sum_i \Delta_e f_i(S_1^e) \leq m$$

which means $\sum_{e\in E} \rho_e \leq 1$ and the proof is completed. $\square$

We then obtain ALG1's performance guarantee as stated in Theorem 6.

Theorem 6. Given an instance of MINRF with input $V, \{f_i\}_{i\in[m]}$ and $r = 1$. If S is an output of ALG1 and S_1 is a $(0, \alpha)$-robust set outputted by ALG0$(V, \emptyset, \{f_i\}_{i\in[m]})$ then $|S| \leq OPT(V, 1)O(|S_1| \ln m + 1/\alpha)$.

Proof. From a ratio of ALG0 and lemma 1, we have $|S_1| \leq O(\ln \frac{m}{\alpha})OPT(V, 0) \leq O(\ln \frac{m}{\alpha})OPT(V, 1)$.

For each $e \in E$ and S^e as defined before, we have:

$$\sum_i f_i(S^e \setminus \{e\}) = \sum_i f_i(S^e) - \rho_e m \geq m(1 - \alpha - \rho_e)$$

The last inequality comes from the fact that $S_1 \subseteq S^e$ and S_1 is $(0, \alpha)$-robust. Then, with $S' =$ ALG0 $(V \setminus \{e\}, S^e \setminus \{e\}, \mathcal{F}')$ in line 4, denote $\delta S^e = S' \setminus S^e$. From the ratio of ALG0 and lemma 1, we have:

$$\begin{aligned} |\delta S^e| &\leq O(\ln \frac{(\alpha + \rho_e)m}{\alpha})OPT(V \setminus \{e\}, 0) \\ &\leq O(\ln \frac{(\alpha + \rho_e)m}{\alpha})OPT(V, 1) \end{aligned}$$

Therefore, with S is the returned solution, we have:

$$\begin{aligned} |S| &= |S_1| + \sum_{e\in E} |\delta S^e| \\ &\leq O\Big(\ln \frac{m}{\alpha} + \sum_{e\in E} \ln \frac{(\alpha + \rho_e)m}{\alpha}\Big)OPT(V, 1) \\ &= O\Big(\ln \frac{m}{\alpha} + \ln \prod_{e\in E} \frac{(\alpha + \rho_e)m}{\alpha}\Big)OPT(V, 1) \\ &\leq O\Big(\ln \frac{m}{\alpha} + \ln \big(\sum_{e\in E} \frac{(\alpha + \rho_e)m}{\alpha|E|}\big)^{|E|}\Big)OPT(V, 1) \\ &\leq O\Big(\ln \frac{m}{\alpha} + \ln \big(m(1 + \frac{1}{\alpha|E|})\big)^{|E|}\Big)OPT(V, 1) \\ &\leq O(|E| \ln m + \frac{1}{\alpha})OPT(V, 1) \end{aligned}$$

Algorithm 3 Algorithm with general r (ALGR)

Input $V, \{f_i\}_{i\in[m]}, r$

```
1: F_0 = {f_i}_{i∈[m]}; S_0 = ALG0 (V, ∅, F_0);
2: for t = 1 → r do
3:     F_t = ∅
4:     for each set X ⊆ S_{t-1} that |X| = r, X ⊈ S_{t-2} do
5:         for each i ∈ [m] s.t. f_i(S_{t-1} \ X) < 1 − α do
6:             Define f_{i,X}(.) = f_i(. \ X)
7:             F_t = F_t ∪ {f_{i,X}}
8:     S_t = ALG0 (V, S_{t-1}, F_t)
```

Return S_r

which completes the proof. $\square$

ALG1's ratio is *tight* by considering a special instance of MINRF, Robust Set Cover with $r = 1$. This tight example is provided in Appendix.

In term of **query complexity**, it is trivial that if ALG1 uses RANDGR or GREEDY as ALG0, each f_i would be queried at most $O(n \max(OPT(V, 1)(|S_1| \ln m + 1/\alpha), n))$ times. If THRESGR is used, each constraint of $\mathcal{F}'$ in line 4 is queried at most $O(n \ln mn)$ times, thus each f_i is queried at most $O(n|S_1| \ln mn)$ times in total.

Algorithm For General r (ALGR)

ALGR works in at most r rounds, in which after t rounds, ALGR guarantees an obtained solution is (t, α)-robust. Denote S_t as the obtained solution after t rounds. At round t, ALGR introduces a new MINRF instance with a new set of functions F_t. Each function in F_t is defined by a function f_i and a set $X \subset S_t$ that $f_i(S_t \setminus X) < 1 - \alpha$ and $|X| = r$. This is a key of ALGR because by solving the new MINRF instance to obtain S_{t+1}, RANDGR guarantees S_{t+1} is (t, α)-robust. Also the algorithm is able to bound the number of newly-added elements in term of $|S^*|$ by observing that S^* is also a feasible solution to the new MINRF instance.

ALGR's pseudocode is presented by Alg. 3. Note that ALGR guarantees S_r is (r, α)-robust without a need of scanning all the removals of its subsets of size r. We prove that by using contradiction as follows:

Assume S_r is not (r, α)-robust, then there exists $X \subseteq V$ and f_i such that $|X| = r$ and $f_i(S_r \setminus X) < 1 - \alpha$. Let $X_0 = X \cap S_0$, and $X_t = X \cap (S_t \setminus S_{t-1})$ for $t = 1 \to r$.

If there exists an empty X_t, let $X' = \cup_{i=0}^{t-1} X_i$. We have $|X'| \leq r$ and $X' \subseteq S_{t-1}$. Due to the output of ALG0 in line 8, $f_i(S_t \setminus X') \geq 1 - \alpha$. But $S_t \setminus X' \subseteq S_r \setminus X$, so $f_i(S_r \setminus X) \geq 1 - \alpha$, which contradicts to our assumption.

Thus, no X_t should be empty, which is impossible since $|X| = r \geq |\cup_{t=0}^{r} X_t|$, and $X_0, ..., X_r$ are disjoint. Therefore, S_r should be (r, α)-robust.

To obtain ALGR's performance guarantee, we have the following lemma.

Lemma 4. $|S_t \setminus S_{t-1}| \leq O(\ln \frac{|F_t|}{\alpha}) OPT(V, r)$ for all $t \leq r$

Proof. Considering a new constraint $f_{i,X}$ created in line 6, it is trivial that the function $f_{i,X}$ is monotone submodular.

Also, as S^* is $(r, 0)$-robust, $f_{i,X}(S^*) = f_i(S^* \setminus X) \geq 1$. That means S^* is feasible for the MINRF instance in line 6, with F_t as a set of constraint and $r = 0$. The lemma follows from the ratio of ALG0. $\square$

Lemma 4 is critical to obtain ALGR's ratio, stated in the following theorem.

Theorem 7. Given an instance of MINRF with input $V, \{f_i\}_{i\in[m]}, r$, if S is an output of ALGR, then:

$$|S| \leq OPT(V, r) O(r \ln m/\alpha + r^2 \ln n)$$

Proof. Using lemma 1 and ALG0's ratio, we have: $|S_0| \leq OPT(V, 0) O(\ln m/\alpha) \leq OPT(V, r) O(\ln m/\alpha)$. Therefore, from lemma 4, we have:

$$\begin{aligned} |S_r| &= |S_0| + \sum_{t=1}^{r} |S_t \setminus S_{t-1}| \\ &\leq O(\ln \frac{m}{\alpha} + \sum_{t=1}^{r} \ln \frac{|F_t|}{\alpha})\ OPT(V, r) \end{aligned}$$

Furthermore, $\sum_{t=1}^{r} |F_t| \leq m\binom{|S_{r-1}|}{r}$ because: (1) No subset $X \in S_{r-1}$ of size r is considered more than one round (line 2) as if $f_i(S_t \setminus X) \leq 1 - \alpha$ then $f_i(S_{t+1} \setminus X) \geq 1 - \alpha$; and (2) each subset X added to F_t at most m new constraints.

Therefore, by using AM-GM inequality, we have: $\prod_t |F_t| \leq (\frac{\sum_t |F_t|}{r})^r \leq (\frac{m}{r}\binom{n}{r})^r$.

Thus, $|S_r| \leq O(r \ln \frac{m}{\alpha} + r^2 \ln n)\ OPT(V, r)$. $\square$

Query Complexity. The bottleneck of ALGR is from the task of finding all subsets X in line 4. As there is $\binom{|S_{r-1}|}{r}$ subsets X, ALGR takes $\binom{|S_{r-1}|}{r}$ queries for each f_i to only find X; and in the worst case, each f_i will generate $\binom{|S_{r-1}|}{r}$ functions $f_{i,X}$ (line 6). Then, if ALGR uses RANDGR or GREEDY as ALG0, in worst case, each f_i is queried at most $O(n(r \ln \frac{m}{\alpha} + r^2 \ln n)\ OPT(V, r)\binom{|S_{r-1}|}{r})$ times. If THRESGR is used, at round t, each f_i is queried at most $O(\frac{n|F_t|}{\gamma} \ln \frac{n|F_t|}{\alpha})$. Overall, ALGR using THRESGR will query each f_i at most $O(\frac{n}{\gamma}\binom{|S_{r-1}|}{r}(r \ln \frac{m}{\alpha} + r^2 \ln n))$ times. ALGR is polynomial with fixed r and favourable if $OPT(V, r) \ll n$.

Experimental Evaluation

In this section, we compare our algorithms with existing methods and intuitive heuristics on two applications of MINRF, Information Propagation for Multiple Groups (IP) and Movie Recommendation for Multiple Users (MR). The source code is available at https://github.com/lannn2410/minrf.

Information Propagation for Multiple Groups (IP) In this problem, a social network is modeled as a directed graph $G = (V, E)$ where V is a set of social users. Each edge (u, v) is associated with a weight $w_{u,v}$, representing the strength of influence from user u to v.

To model the information propagation process, we use Linear Threshold (LT) Model (Kempe, Kleinberg, and Tardos

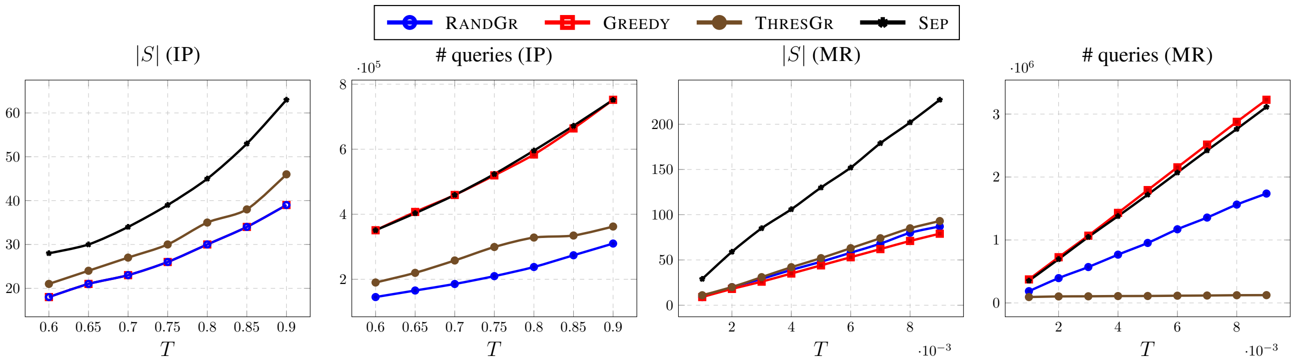

Figure 1: Performance of algorithms with $r = 0$

2003; Nguyen and Thai 2020). In general, the process is as follows: Each $v \in V$ has a threshold θ_v chosen uniformly at random in $[0, 1]$ and the information start from a seed set $S \subseteq V$. At first all users in S become active. Next, information cascades in discrete steps and in each step, a user v becomes active if $\sum_{\text{active } u} w_{u,v} \geq \theta_v$. The process stops when no more user can become active.

Given a collection $\mathcal{U}$ of subsets of V, i.e $\mathcal{U} = \{C_1, ..., C_m\}$ where $C_i \subseteq V$. Each C_i represents a group that we need to influence. Denote $I_i(S)$ as the expected number of active users in C_i by a seed set S. Given a number $T \in [0, 1]$, IP aims to find the smallest S such that for all $C_i \in \mathcal{U}$, $\min_{|X| \leq r} I_i(S \setminus X) \geq T|C_i|$.

We use Facebook dataset from SNAP database (Leskovec and Krevl 2014), an undirected graph with 4,039 nodes and 88,234 edges. Since it is undirected, we treat each edge as two directed edges. The weight $w_{u,v}$ is set to be $1/d_v$ where d_v is in-degree of v. $\mathcal{U}$ is a collection of groups to which users are classified based on their gender or race. Due to lack of data information, a user's race and gender are randomly assigned. $I_i(S)$ is estimated over 100 graph samples.

Movie Recommendation for Multiple Users (MR) In this problem, given a set M of movies, a set U of users, each user u has a list L_u of his/her favourite movies. Given $S \subseteq U$, a utility score of u to S is defined as $f_u(S) = \sum_{i \in L_u; j \in S \setminus L_u} s_{i,j}$ (Mirzasoleiman, Zadimoghaddam, and Karbasi 2016) where $s_{i,j} \in [0, 1]$ which measures the similarity between movie i and j. Given a number T, the objective is to find the smallest set of movies to recommend to all users in a way such that every user's utility level is at least T under any r "inaccurate-data" movies removal, i.e. $\min_{|X| \leq r} f_u(S \setminus X) \geq T$ for all $u \in U$.

We use Movie Lens dataset from GroupLens (2015) database, which includes information of 10,381 movies; and their 20,000,264 ratings (ranging in $[0, 5]$) from 138,493 users. We randomly pick 4 users for a set U, L_u contains movies that u rated at least 4. Each movie i is associated by a 1,129-dimension vector v_i, where each entry (ranging in $[0, 1]$) represents the relevant score between the movie and a keyword. The relevant scores are available in the dataset. We use cosine similarity score $\frac{v_i \cdot v_j}{\|v_i\| \|v_j\|}$ to present $s_{i,j}$. For each user u, $f_u(S)$ is normalized to be in range $[0, 1]$.

Compared Algorithms With $r = 0$, we compare RandGr, Greedy and ThresGr ($\gamma = 0.2$) with Sep algorithm: which considers each constraint separately, runs greedy to find a set S_i that $f_i(S_i) \geq 1 - \alpha$ and return $\cup_{i \in [m]} S_i$. Sep obtains a ratio of $O(m \ln \frac{1}{\alpha})$.

With $r > 0$, we compare AlgR's performance in combination with each Alg0, including RandGr, Greedy, ThresGr, Sep. Each combination of AlgR to a Alg0 algorithm is denoted, in short, AlgR-name of the Alg0 algorithm, e.g. AlgR-RandGr.

We also compare AlgR with DisJoint, a heuristic we propose to evaluate. DisJoint finds $r + 1$ disjoint sets $S_1, ..., S_{r+1}$ such that $f_i(S_j) \geq 1 - \alpha$ for all $i \in [m]$ and $j \in [r + 1]$; and returns $S = \cup_{j \in [r+1]} S_j$. If DisJoint successfully finds all $\{S_j\}_{j \in [r+1]}$, then S is feasible to MinRF without the need for checking all subsets of size r. This is because for any set X of size r, there should exist S_j that $S_j \cap X = \emptyset$. Thus, $S_j \subset S \setminus X$, which means $f_i(S \setminus X) \geq f_i(S_j) \geq 1 - \alpha$ for all $i \in [m]$. However, there are two problems with DisJoint: (1) If DisJoint cannot find all $\{S_j\}_{j \in [r+1]}$, the algorithm does not guarantee there exists no feasible solution to MinRF; and (2) DisJoint does not obtain any approximation ratio.

For $r = 1$, we also evaluate Alg1 performance in combination with each Alg0 algorithm, including RandGr, Greedy, ThresGr.

Other. We set $\alpha = 0.1$. Results are averaged over 10 repetitions.

Experimental Results

Fig. 1 shows the performances of different Alg0 algorithms in comparison with Sep. We can see that Alg0 algorithms totally outperformed Sep in solution quality by a huge margin. RandGr returned solutions approximately close to Greedy, which is the best one in term of solution quality. However, in term of query efficiency, RandGr took much fewer queries than Greedy and; and was the fastest algorithm in the IP problem. This confirms the efficiency of RandGr by introducing randomness and discarding satisfied constraints after each iteration.

Fig. 2 shows algorithms' performance on the IP and MR problems when $r = 1$. The two proposed heuristics, Alg1-Sep and DisJoint, showed the worst performance in solu-

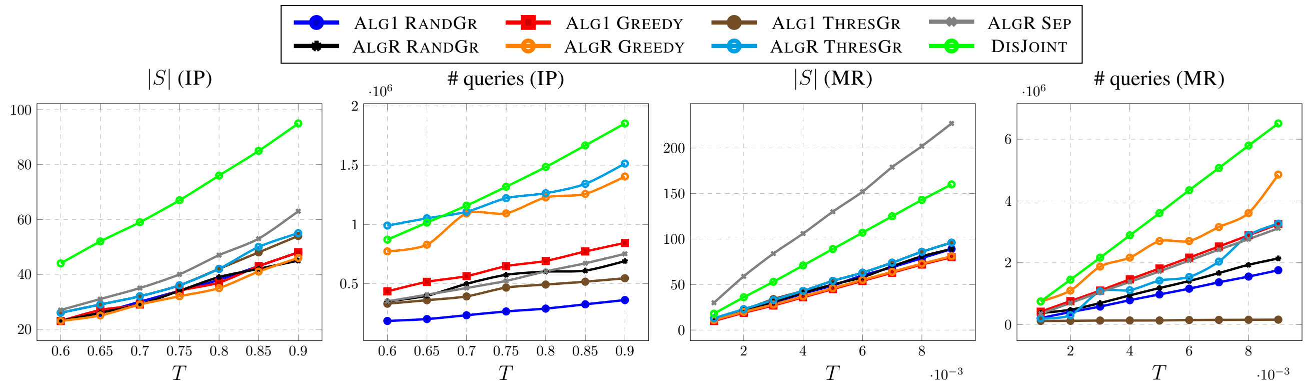

Figure 2: Performance of algorithms with $r = 1$

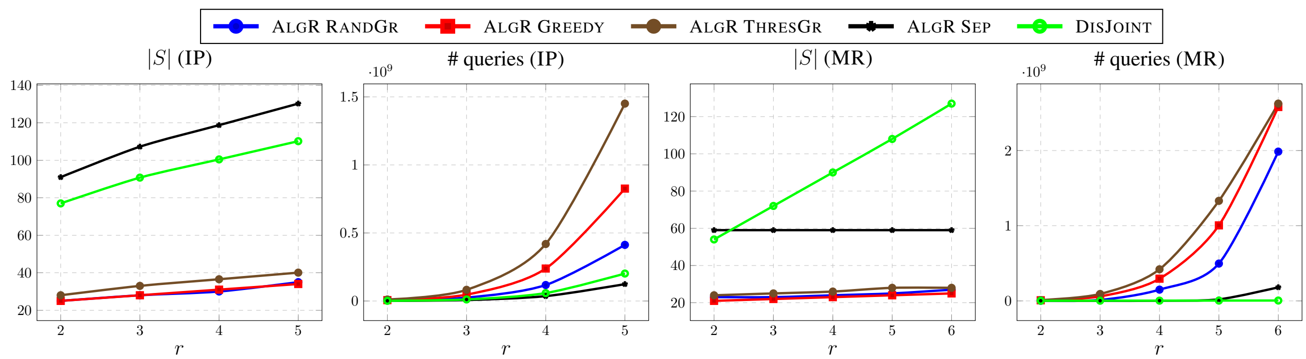

Figure 3: Performance of algorithms with various r

tion quality. DISJOINT's undesirable performance came from the fact that a union of disjoint subsets, each is able to satisfy all constraints, is not a necessary condition to guarantee robustness. Also, by finding disjoint subsets, DISJOINT needed more queries than any other algorithms.

In combination with the same ALG0 algorithm, ALG1 and ALGR had almost similar returned solution but ALG1 totally outperformed ALGR in term of number of queries. That can be explained by the fact that whenever ALG1 finds an element e whose removal violates at least one constraint, ALG1 will add elements to compensate for e's removal. That guarantees not only S is robust to e's removal but also the newly-added elements may help S being robust against some other elements' removal as well. On the other hand, ALGR gathers all elements, each element's removal violates at least one constraint, to form a new MINRF instance with a much larger set of submodular functions than ALG1. That helps ALG1 obtain better number of queries than ALGR.

Fig. 3 shows algorithms' performance with larger r. We observed that ALGR-SEP and DISJOINT were outperformed by other algorithms by a huge margin in solution quality; but took much fewer number of queries than the others. That is because with larger r, the number of subsets of size r is increased by an exponent rate in term of r, which increases significantly the number of queries of ALGR for scanning subsets of size r of S_{r-1}. SEP was less suffered than our ALG0 algorithms because SEP returned much larger solutions, which can reach robustness at S_t where $t \ll r$. On the other hand, DISJOINT had the small number of queries because DISJOINT does not need to scan all removals of its subsets of size r to check feasibility of the returned solution.

Fig. 3 also shows that: ALGR-RANDGR performed the best in both solution quality and the number of queries in comparison with ALGR-GREEDY and ALGR-THRESGR. Although THRESGR was the most efficient ALG0 algorithm when $r = 0$ (standalone) or $r = 1$ (combining with ALG1 or ALGR), ALGR-THRESGR's performances were undesirable with large r. This is because THRESGR tends to return larger solution than RANDGR and GREEDY. Therefore, ALGR-THRESGR requires more queries to scan over all subset of size r of S_{r-1} than ALGR-RANDGR and ALGR-GREEDY.

Conclusion

Motivated by real-world applications, in this work, we studied a problem of minimum robust set subject to multiple submodular constraints, namely MINRF. We investigate MINRF's hardness using complexity theories; and proposed multiple approximation algorithms to solve MINRF. Our algorithms are proven to return tight performance guarantees to MINRF's inapproximability and required query complexity. Finally, we empirically demonstrated that our algorithms outperform several intuitive methods in terms of the solution quality and number of queries.

Acknowledgements

This work was supported in part by the National Science Foundation (NSF) grants IIS-1908594, IIS-1939725, and the University of Florida Informatics Institute Fellowship Program. We would like to thank the anonymous reviewers for their helpful feedback.

References

Anari, N.; Haghtalab, N.; Naor, S.; Pokutta, S.; Singh, M.; and Torrico, A. 2019. Structured Robust Submodular Maximization: Offline and Online Algorithms. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 3128–3137.

Bogunovic, I.; Mitrović, S.; Scarlett, J.; and Cevher, V. 2017. Robust submodular maximization: A non-uniform partitioning approach. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 508–516. JMLR. org.

Dinh, T. N.; and Thai, M. T. 2014. Network under joint node and link attacks: Vulnerability assessment methods and analysis. *IEEE/ACM Transactions on Networking* 23(3): 1001–1011.

Goyal, A.; Bonchi, F.; Lakshmanan, L. V.; and Venkatasubramanian, S. 2013. On minimizing budget and time in influence propagation over social networks. *Social network analysis and mining* 3(2): 179–192.

GroupLens. 2015. MovieLens 20M Dataset. https://grouplens.org/datasets/movielens/20m/.

Guillory, A.; and Bilmes, J. A. 2011. Simultaneous learning and covering with adversarial noise .

Iyer, R. K.; and Bilmes, J. A. 2013. Submodular optimization with submodular cover and submodular knapsack constraints. In *Advances in Neural Information Processing Systems*, 2436–2444.

Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 137–146. ACM.

Krause, A.; McMahan, H. B.; Guestrin, C.; and Gupta, A. 2008. Robust submodular observation selection. *Journal of Machine Learning Research* 9(Dec): 2761–2801.

Kuhnle, A.; Pan, T.; Alim, M. A.; and Thai, M. T. 2017. Scalable bicriteria algorithms for the threshold activation problem in online social networks. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, 1–9. IEEE.

Leskovec, J.; and Krevl, A. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

Mirzasoleiman, B.; Karbasi, A.; Badanidiyuru, A.; and Krause, A. 2015. Distributed submodular cover: Succinctly summarizing massive data. In *Advances in Neural Information Processing Systems*, 2881–2889.

Mirzasoleiman, B.; Zadimoghaddam, M.; and Karbasi, A. 2016. Fast distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*, 3594–3602.

Mitrovic, S.; Bogunovic, I.; Norouzi-Fard, A.; Tarnawski, J. M.; and Cevher, V. 2017. Streaming robust submodular maximization: A partitioned thresholding approach. In *Advances in Neural Information Processing Systems*, 4557–4566.

Nguyen, H. T.; Thai, M. T.; and Dinh, T. N. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *Proceedings of the 2016 International Conference on Management of Data*, 695–710. ACM.

Nguyen, L.; and Thai, M. T. 2020. Streaming k-Submodular Maximization under Noise subject to Size Constraint. In *International Conference on Machine Learning*, 7338–7347. PMLR.

Nguyen, L. N.; and Thai, M. T. 2019. Network Resilience Assessment via QoS Degradation Metrics: An Algorithmic Approach. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 3(1): 1–32.

Nguyen, L. N.; Zhou, K.; and Thai, M. T. 2019. Influence maximization at community level: A new challenge with non-submodularity. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 327–337. IEEE.

Norouzi-Fard, A.; Bazzi, A.; Bogunovic, I.; El Halabi, M.; Hsieh, Y.-P.; and Cevher, V. 2016. An efficient streaming algorithm for the submodular cover problem. In *Advances in Neural Information Processing Systems*, 4493–4501.

Ohsaka, N.; and Yoshida, Y. 2015. Monotone k-submodular function maximization with size constraints. In *Advances in Neural Information Processing Systems*, 694–702.

Orlin, J. B.; Schulz, A. S.; and Udwani, R. 2018. Robust monotone submodular function maximization. *Mathematical Programming* 172(1-2): 505–537.

Qian, C.; Shi, J.-C.; Yu, Y.; Tang, K.; and Zhou, Z.-H. 2017. Subset selection under noise. In *Advances in neural information processing systems*, 3560–3570.

Staib, M.; Wilder, B.; and Jegelka, S. 2019. Distributionally Robust Submodular Maximization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 506–516.

Zhang, H.; Mishra, S.; Thai, M. T.; Wu, J.; and Wang, Y. 2014. Recent advances in information diffusion and influence maximization in complex social networks. *Opportunistic Mobile Social Networks* 37(1.1): 37.