Fair Representation in Submodular Subset Selection: A Pareto Optimization Approach

Anonymous authors
Paper under double-blind review

Abstract

Many machine learning applications, such as feature selection, recommendation, and social advertising, require the joint optimization of the global *utility* and the *representativeness* for different groups of items or users. We thus propose a novel multi-objective combinatorial optimization problem called *Submodular Maximization with Fair Representation* (SMFR), which selects subsets from a ground set, subject to a knapsack or matroid constraint, so as to maximize a submodular (*utility*) function f, while a set of d submodular (*representativeness*) functions $g_1, \dots, g_d$ are also maximized. We show that the maximization of f might conflict with the maximization of $g_1, \dots, g_d$, so that no single solution can optimize all of them at the same time. Therefore, we propose a Pareto optimization approach to SMFR, which finds a set of solutions to approximate all Pareto optimal solutions with different trade-offs between these objectives. Our method converts an instance of SMFR into several submodular cover instances by adjusting the weights of objective functions; then it computes a set of solutions by running the greedy algorithm on each instance. We prove that our method provides approximation guarantees for SMFR under knapsack or matroid constraints. Finally, we demonstrate the effectiveness of SMFR and our proposed approach in two real-world problems: *maximum coverage* and *recommendation*.

1 Introduction

The problem of subset selection aims to pick a maximum utility subset S, under a given constraint, from a ground set V of items. This fundamental problem arises in a wide range of machine learning applications, such as viral marketing and social advertising (Kempe et al., 2003; Aslay et al., 2015; 2017; Tang, 2018), recommendation systems (Tschiatschek et al., 2017; Mehrotra & Vishnoi, 2023), data summarization (Lin & Bilmes, 2010; Mirzasoleiman et al., 2016), and feature selection (Liu et al., 2013; Bao et al., 2022), to name just a few. A common combinatorial structure in such problems is *submodularity* (Krause & Golovin, 2014), which naturally captures the “diminishing returns” property: adding an item to a smaller set produces a higher marginal gain than adding it to a larger set. This property not only captures the desirable properties of *coverage* and *diversity* of subsets, but also enables the design of efficient approximation algorithms.

Among the various combinatorial optimization problems for subset selection in the literature, maximizing a monotone submodular function subject to a knapsack constraint (SMK) or a matroid constraint (SMM) has attracted a lot of attention, as such constraints capture common scenarios in which the selected subset must be limited within a budget (Nemhauser & Wolsey, 1978; Fisher et al., 1978; Sviridenko, 2004; Krause & Guestrin, 2005; Vondrak, 2008; Călinescu et al., 2011; Badanidiyuru & Vondrák, 2013; Filmus & Ward, 2014; Buchbinder et al., 2019; Ene & Nguyen, 2019a;b; Huang et al., 2020; Yaroslavtsev et al., 2020; Tang et al., 2021; Han et al., 2021; Feldman et al., 2022; Li et al., 2022).

More formally, given a ground set V of n items, we consider a set function $f : 2^V \to \mathbb{R}^+$ to measure the *utility* $f(S)$ of any set $S \subseteq V$. We assume that f is normalized, i.e., $f(\emptyset) = 0$, monotone, i.e., $f(S) \leq f(T)$ for any $S \subseteq T \subseteq V$, and submodular, $f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$ for any $S \subseteq T \subseteq V$ and $v \in V \setminus T$. We also consider a cost function $c : V \to \mathbb{R}^+$ which assigns a positive cost $c(v)$ to each item $v \in V$, and we denote $c(S)$ the cost of a set $S \subseteq V$, defined as the sum of costs for all items in S, i.e., $c(S) = \sum_{v \in S} c(v)$.

For a given budget $k \in \mathbb{R}^+$, the set of all feasible solutions subject to the knapsack constraint contains all subsets of V whose costs are at most k, i.e., $\mathcal{I}_k = \{S \subseteq V : c(S) \leq k\}$. The SMK problem on f is thus defined as $S_f^* = \arg\max_{S \in \mathcal{I}_k} f(S)$. Furthermore, a matroid $\mathcal{M}$ on a ground set V is defined by a collection $\mathcal{I}(\mathcal{M})$ of subsets of V called the *independent sets*, that satisfies the following properties: (1) $\emptyset \in \mathcal{I}(\mathcal{M})$; (2) for any $S \subset T \subseteq V$, if $T \in \mathcal{I}$, then $S \in \mathcal{I}(\mathcal{M})$ holds; (3) for any $S, T \subseteq V$, if $|S| < |T|$, there exists $v \in T \setminus S$ such that $S \cup \{v\} \in \mathcal{I}(\mathcal{M})$. Here, the size of the maximum independent sets in $\mathcal{M}$ is called its rank $r(\mathcal{M})$. Similarly to SMK, the SMM problem on f is defined as $S_f^* = \arg\max_{S \in \mathcal{I}(\mathcal{M})} f(S)$.

In many real-world problems, in addition to the primary objective of maximizing the utility function f, it is often essential to take into account the representativeness with respect to different groups of items or users. For example, consider the following influence maximization problem (Tsang et al., 2019; Becker et al., 2020):

Example 1. *Let $\mathcal{G} = (V, E)$ be a graph that denotes the relationships between a set of users V on a social network. Each user $v \in V$ is also associated with a sensitive attribute $\mathcal{A}$ to divide V into multiple protected groups. The influence maximization (IM) problem (Kempe et al., 2003) aims to select a subset $S \subseteq V$ of users as seeds to maximize a (monotone, submodular) influence spread function under an information diffusion (e.g., independent cascade or linear threshold) model. If the information to be spread is related to education and employment opportunities, fair access to information between protected groups (Tsang et al., 2019; Becker et al., 2020) becomes a critical issue. This is often formulated as maximizing the influence spread functions specific to all protected groups in a balanced manner so that none of the groups is much worse off than the others. Furthermore, constraints in different contexts can be imposed on the seed set, e.g., to limit the overall budget for the propagation campaign, the cost of the seeds should be within an upper bound (knapsack constraint), or to achieve a fair representation at the group level, the number of seeds selected from each group cannot exceed an upper limit (matroid constraint).*

The above problem, as well as many other subset selection problems with fairness or other representativeness considerations (Krause et al., 2008; Mirzasoleiman et al., 2016; Wang et al., 2024), can be formulated as a multi-objective optimization problem of maximizing a monotone submodular *utility* function f and a set of d monotone submodular *representativeness* functions $g_1, \ldots, g_d$, all defined on the same ground set V, subject to a knapsack or matroid constraint:

$$\arg\max_{S \in \mathcal{I}} \big(f(S), g_1(S), \ldots, g_d(S)\big). \tag{1}$$

We call this problem *Submodular Maximization with Fair Representation* (SMFR) since it captures the case where the submodular utility function is maximized while all the submodular representativeness functions are also maximized to avoid under-representing any of them.

Our Contributions. To the best of our knowledge, SMFR is a novel optimization problem, never addressed before (see Section 2 for a detailed discussion of how the related literature differs from SMFR). It is easy to see that SMFR is at least as hard as SMK and SMM, which cannot be approximated within a factor better than $1 - 1/e$ unless $P = NP$ (Feige, 1998; Khuller et al., 1999). However, SMFR is much more challenging than SMK and SMM due to its multi-objective nature. By providing a counterexample, we show that there might not exist any single solution to an instance of SMFR that achieves an approximation factor greater than 0 to maximize f and $g_1, \ldots, g_d$ simultaneously, even for a special case of $d = 1$. As such, we consider approaching SMFR in Eq. 1 by *Pareto optimization*. Specifically, we call a set S an (α, β)-approximate solution for an instance of SMFR if $S \in \mathcal{I}$, $f(S) \geq \alpha \mathtt{OPT}_f$, where $\mathtt{OPT}_f = \max_{S' \in \mathcal{I}} f(S')$, and $g_i(S) \geq \beta \mathtt{OPT}_{g_i}$ for all $i = 1, \ldots, d$, where $\mathtt{OPT}_{g_i} = \max_{S' \in \mathcal{I}} g_i(S')$. An (α, β)-approximate solution S is Pareto optimal if there does not exist any (α', β')-approximate solution for any $\alpha' \geq \alpha, \beta' \geq \beta$ (and at least one is strictly larger). Since computing any Pareto optimal solution to SMFR is still NP-hard, we propose a general framework to find a set of solutions to approximate the *Pareto frontier* consisting of all Pareto optimal solutions. Our framework first uses any existing algorithm for SMK (Sviridenko, 2004; Yaroslavtsev et al., 2020; Tang et al., 2021; Feldman et al., 2022; Li et al., 2022) or SMM (Fisher et al., 1978; Vondrak, 2008; Călinescu et al., 2011; Badanidiyuru & Vondrák, 2013; Filmus & Ward, 2014; Buchbinder et al., 2019) to approximate $\mathtt{OPT}_f$ and each $\mathtt{OPT}_{g_i}$. Based on the approximations, our proposal transforms an instance of SMFR into multiple instances of the submodular cover problem with different weights on $\mathtt{OPT}_f$ and each

$\texttt{OPT}_{g_i}$ to capture the trade-offs between f and each g_i. Then, classic greedy algorithms (Wolsey, 1982; Torrico et al., 2021) are used to obtain an approximate solution for each submodular cover instance. Finally, all the above-computed solutions that are not "dominated"[1] by any other solution are returned as the set $\mathcal{S}$ of at most $O(\frac{1}{\varepsilon})$ approximate solutions to SMFR for any $\varepsilon \in (0, 1)$. Theoretically, our framework provides approximation bounds for SMFR under both knapsack and matroid constraints:

- When using a δ-approximation algorithm for SMK, it provides a set $\mathcal{S}$ such that for any (α, β)-approximate Pareto optimal solution of SMFR, there must exist a corresponding $(\delta\alpha - \varepsilon, \delta\beta - \varepsilon)$-approximate solution of cost $O(k \log \frac{d}{\varepsilon})$ in $\mathcal{S}$, where $k \in \mathbb{R}^+$ is the budget of the knapsack constraint.

- When using a δ-approximation algorithm for SMM, it also provides a set $\mathcal{S}$ such that for any (α, β)-approximate Pareto optimal solution of SMFR, there must exist a corresponding $(\delta\alpha - \varepsilon, \delta\beta - \varepsilon)$-approximate solution of size $O(r \log \frac{d}{\varepsilon})$ in $\mathcal{S}$, where $r \in \mathbb{Z}^+$ is the rank of the matroid constraint.

In our empirical assessment, we evaluate our proposed framework through extensive experiments on the problems of *maximum coverage* and *recommendation* using real-world data. The numerical results confirm the effectiveness of our proposal compared to competitive baselines.

Paper Organization. The rest of this paper is organized as follows. We review the related work in Section 2. Then, we analyze the hardness of SMFR in Section 3. Next, our algorithmic framework for SMFR is presented in Section 4. Subsequently, the experimental setup and results are provided in Section 5. Finally, we conclude the paper and discuss future work in Section 6. The proofs of theorems and lemmas and several supplemental experiments are deferred to the appendices due to space limitations.

2 Related Work

Monotone Submodular Maximization with Knapsack or Matroid Constraints. There exists a wide literature on maximizing a monotone submodular function subject to a knapsack constraint (SMK). For cardinality constraints, a special case of both knapsack and matroid constraints, Nemhauser et al. (1978) proposed a simple greedy algorithm that runs in $O(kn)$ time and yields the best possible approximation factor $1 - 1/e$ unless $P = NP$. However, the greedy algorithm can be arbitrarily bad for general knapsack or matroid constraints. Sviridenko (2004) first proposed a greedy algorithm with partial enumerations that achieves the best possible approximation $1 - 1/e$ for SMK in $O(n^5)$ time. Kulik et al. (2021) and Feldman et al. (2022) improved the time complexity to $O(n^4)$ while keeping the same approximation factor. Krause & Guestrin (2005) proposed an $O(n^2)$-time $\frac{1}{2}(1 - \frac{1}{e}) \approx 0.316$-approximation cost-effective greedy algorithm for SMK. Tang et al. (2021), Kulik et al. (2021), and Feldman et al. (2022) improved the approximation factor of the cost-effective greedy algorithm to 0.405, $[0.427, 0.4295]$, and $[0.427, 0.462]$ independently. Ene & Nguyen (2019a) proposed a near-linear time $(1 - 1/e - \varepsilon)$-approximation algorithm for SMK based on multilinear relaxation. Yaroslavtsev et al. (2020) proposed a $\frac{1}{2}$-approximation Greedy+Max algorithm for SMK in $O(n^2)$ time. Feldman et al. (2022) further provided an approximation factor of 0.6174 in $O(n^3)$ time by enumerating each item as a partial solution and running Greedy+Max on each partial solution. Li et al. (2022) recently proposed a $(\frac{1}{2} - \varepsilon)$-approximation algorithm for SMK in $O(\frac{n}{\varepsilon} \log \frac{1}{\varepsilon})$ time.

Maximizing a monotone submodular function subject to a matroid constraint (SMM) has also been extensively investigated. Fisher et al. (1978) first proposed a $\frac{1}{2}$-approximation greedy algorithm for SMM running in $O(nr)$ time. Călinescu et al. (2011) and Vondrak (2008) independently proposed randomized continuous greedy algorithms with rounding for SMM. Both algorithms achieved the best possible $(1-1/e)$-approximation in expectation but had prohibitive $O(n^8)$ running time. Badanidiyuru & Vondrák (2013) proposed a faster continuous greedy algorithm that yielded a $(1 - 1/e - \varepsilon)$-approximation for SMM in $O(\frac{n^2}{\varepsilon^4} \log^2 \frac{n}{\varepsilon})$ time. Filmus & Ward (2014) proposed a $(1 - 1/e - \varepsilon)$-approximation algorithm in $O(\frac{nr^4}{\varepsilon^3})$ time and a $(1 - 1/e)$-approximation algorithm in $O(n^2 r^7)$ time, both randomized and based on non-oblivious local search. Buchbinder et al. (2019) proposed the first deterministic algorithm for SMM with an approximation

---

[1] A solution S will be dominated by another solution T if the approximation factors α, β of S are both no greater than those of T and at least one is strictly smaller.

factor over 1/2 in $O(nr^2)$ time. Ene & Nguyen (2019b) also proposed a nearly-linear time $(1 - 1/e - \varepsilon)$-approximation algorithm for SMM based on multilinear relaxation. Although the above algorithms cannot be applied directly to SMFR, any of them can serve as a subroutine in our algorithmic framework for SMFR.

Multi-objective Submodular Maximization. There exist also several variants of submodular maximization problems to deal with more than one objective. We next consider only multi-objective submodular maximization problems that are relevant to SMFR. The problem of maximizing the minimum of $d > 1$ submodular functions $g_1, \ldots, g_d$ was studied in (Krause et al., 2008; Udwani, 2018; Anari et al., 2019; Torrico et al., 2021). This problem differs from SMFR because it does not consider maximizing f and aims to return only a single solution for all functions. Nevertheless, we draw inspiration from the SATURATE framework first proposed by Krause et al. (2008) to solve SMFR. Another two relevant problems to SMFR are *Submodular Maximization under Submodular Cover* (SMSC) (Ohsaka & Matsuoka, 2021), which maximizes one submodular function subject to the value of the other submodular function not being below a threshold, and *Balancing utility and fairness in Submodular Maximization* (BSM) (Wang et al., 2024), which maximizes a submodular utility function subject to that a fairness function in form of the minimum of $d > 1$ submodular functions is approximately maximized. SMSC and BSM differ from SMFR in the following four aspects: (*i*) they still return a single solution to optimize a user-specified trade-off between multiple objectives; (*ii*) they are specific to cardinality constraints but cannot handle more general knapsack or matroid constraints; (*iii*) SMSC is limited to two submodular functions, i.e., a special case of $d = 1$ in SMFR; (*iv*) BSM requires all objective functions to be decomposable. Thus, SMFR can work in more general scenarios than SMSC and BSM. Due to the above differences, the algorithms for SMSC and BSM cannot be used for SMFR, and they will be compared to our algorithm after adaptations in the experiments. Very recently, Tang & Yuan (2023) proposed a randomized subset selection method to maximize a (submodular) overall utility function while the (submodular) utility functions for d groups are all not below a lower bound in expectation. They also considered the problem of submodular maximization with group equality, which ensures that the difference in the utilities of any two groups is As they limit their consideration to cardinality constraints and their problem formulations are different from SMFR, their proposed methods are not applicable to SMFR. The problem of regret-ratio minimization (Soma & Yoshida, 2017; Feng & Qian, 2021; Wang et al., 2023) for multi-objective submodular maximization is similar to SMFR in the sense that they also aim to find a set of approximate solutions for different trade-offs between multiple objectives. However, they consider denoting the trade-offs as different non-negative linear combinations of multiple submodular functions but cannot guarantee any approximation for each objective individually.

Finally, several subset selection problems, e.g., (Qian et al., 2015; 2017; 2020; Roostapour et al., 2022), utilize a Pareto optimization method by transforming a single-objective problem into a bi-objective problem and then solving the bi-objective problem to obtain a solution to the original problem. These problems are interesting but orthogonal to our work.

3 Hardness of SMFR

In this paper, we focus on the *Submodular Maximization with Fair Representation* (SMFR) problem in Eq. 1 subject to a knapsack or matroid constraint. Next, we formally analyze the theoretical hardness of SMFR. Since SMK and SMM are both NP-hard and cannot be approximated within a factor $1 - 1/e + \varepsilon$ in polynomial time for any $\varepsilon > 0$ unless $P = NP$ (Feige, 1998; Khuller et al., 1999), the problem of maximizing f or each g_i individually can only be solved approximately. We provide a trivial example to indicate that the maximization of f and the maximization of each g_i could conflict with each other, and there might not exist any $S \in \mathcal{I}$ with approximation factors greater than 0 for both of them, even when $d = 1$.

Example 2. *Suppose that $d = 1$ and the set of feasible solutions $\mathcal{I}$ is defined by a cardinality constraint 1, i.e., $\mathcal{I} = \{S \subseteq V : |S| \leq 1\}$. Note that a cardinality constraint is a special case of both knapsack and matroid constraints. For the two functions f and g_1, we have $\texttt{OPT}_f = f(\{v_0\}) = 1$, $\texttt{OPT}_{g_1} = g_1(\{v_1\}) = 1$, $g_1(\{v_0\}) = 0$, $f(\{v_1\}) = 0$, and $f(\{v_j\}) = g_1(\{v_j\}) = 0$ for any $j > 1$. In the above SMFR instance, there is no set $S \in \mathcal{I}$ such that $f(S) > 0$ and $g_1(S) > 0$.*

Given the above result, we are motivated to introduce *Pareto optimization*, a well-known concept for multi-objective optimization (Qian et al., 2015; Soma & Yoshida, 2017) which provides more than one solution with different (best possible) trade-offs between multiple objectives. We call a set $S \in \mathcal{I}$ an (α, β)-approximate solution for an instance of SMFR if $f(S) \geq \alpha \mathtt{OPT}_f$ and $g_i(S) \geq \beta \mathtt{OPT}_{g_i}$ for each $i \in [d]$. An (α, β)-approximate solution S is Pareto optimal if there does not exist any (α', β')-approximate solution for $\alpha' \geq \alpha$ and $\beta' \geq \beta$ (and at least one is strictly larger). Ideally, by enumerating all distinct Pareto optimal solutions (which form the so-called *Pareto frontier*), one can obtain all different optimal trade-offs between maximizing f and each g_i. However, computing any Pareto optimal solution is still NP-hard. To circumvent the barrier, a feasible approach to SMFR is to find a set $\mathcal{S}$ of approximate solutions, in which, for any Pareto optimal solution, at least one solution close to it is included. This is the approach we follow in our framework.

4 The SMFR-Saturate Framework

To find approximate solutions to an instance of SMFR, we propose to transform it into a series of instances of its corresponding decision problems, that is, to determine whether there exists any (α, β)-approximate solution for the SMFR instance. Then, we introduce the SATURATE framework first proposed in (Krause et al., 2008) to approximately solve each instance of the decision problem as *Submodular Cover* (SC), that is, the problem of finding a set S_c^* with the minimum cardinality/cost such that $f(S_c^*) \geq L$ for some $L \in \mathbb{R}^+$. Now, we formally define the decision problem and analyze why the transformation follows.

Definition 1 (SMFR-DEC)**.** Given an instance of SMFR and two approximation factors $\alpha, \beta \in [0, 1]$, find a set $S \in \mathcal{I}_k$ such that $f(S) \geq \alpha \mathtt{OPT}_f$ and $g_i(S) \geq \beta \mathtt{OPT}_{g_i}$ for each $i \in [d]$, or decide that there does not exist any set that can meet the conditions.

Assuming that $\mathtt{OPT}_f$ and each $\mathtt{OPT}_{g_i}$ are already known, the above conditions can be equivalently expressed as $\frac{f(S)}{\alpha \mathtt{OPT}_f} \geq 1$ and $\frac{g_i(S)}{\beta \mathtt{OPT}_{g_i}} \geq 1$. Then, using the truncation technique in (Krause et al., 2008), SMFR-DEC is converted to decide whether the objective value of the following problem is $d + 1$:

$$\max_{S \in \mathcal{I}} F_{\alpha,\beta}(S) := \min\Big\{1, \frac{f(S)}{\alpha \mathtt{OPT}_f}\Big\} + \sum_{i=1}^{d} \min\Big\{1, \frac{g_i(S)}{\beta \mathtt{OPT}_{g_i}}\Big\}. \tag{2}$$

Note that $F_{\alpha,\beta}$ is ill-formulated due to division by zero when α, β or $\mathtt{OPT}_f$, $\mathtt{OPT}_{g_i}$ are equal to 0. To solve this problem, the first term of $F_{\alpha,\beta}$ is replaced by 1 when $\alpha = 0$ or $\mathtt{OPT}_f = 0$; the second term of $F_{\alpha,\beta}$ is replaced by d when $\beta = 0$ or $\mathtt{OPT}_{g_i} = 0$ for any $i \in [d]$.

The above conversion holds because $F_\alpha(S) = d+1$ if and only if $f(S) \geq \alpha \mathtt{OPT}_f$ and $g_i(S) \geq \beta \mathtt{OPT}_{g_i}$, $\forall i \in [d]$. In addition, $F_{\alpha,\beta}$ is a normalized, monotone, and submodular function because the minimum of a positive real number and a monotone submodular function is monotone and submodular (Krause et al., 2008), and the nonnegative linear combination of monotone submodular functions is monotone and submodular (Krause & Golovin, 2014). In this way, SMFR-DEC is transformed to SC on $F_{\alpha,\beta}$.

Since computing $\mathtt{OPT}_f$ and $\mathtt{OPT}_{g_i}$ is NP-hard, we should use any existing algorithm for SMK (Sviridenko, 2004; Yaroslavtsev et al., 2020; Tang et al., 2021; Feldman et al., 2022; Li et al., 2022) or SMM (Fisher et al., 1978; Vondrak, 2008; Călinescu et al., 2011; Badanidiyuru & Vondrák, 2013; Filmus & Ward, 2014; Buchbinder et al., 2019) to compute their approximations. Suppose that we run an approximation algorithm for SMK or SMM to obtain $\mathtt{OPT}'_f \leq \mathtt{OPT}_f$ and $\mathtt{OPT}'_{g_i} \leq \mathtt{OPT}_{g_i}$, $\forall i \in [d]$ accordingly. The problem in Eq. 2 is relaxed as follows:

$$\max_{S \in \mathcal{I}} F'_{\alpha,\beta}(S) := \min\Big\{1, \frac{f(S)}{\alpha \mathtt{OPT}'_f}\Big\} + \sum_{i=1}^{d} \min\Big\{1, \frac{g_i(S)}{\beta \mathtt{OPT}'_{g_i}}\Big\}, \tag{3}$$

where the problem of division by zero is solved in the same way as for $F_{\alpha,\beta}$ when α, β or $\mathtt{OPT}'_f$, $\mathtt{OPT}'_{g_i}$ are equal to 0. Next, the following lemma indicates that SMFR-DEC can still be answered approximately by solving the relaxed problem in Eq. 3.

Lemma 1. *If $F'_{\alpha,\beta}(S) \geq d + 1 - \frac{\varepsilon}{2}$ for any set $S \in \mathcal{I}$, then S is a $(\delta\alpha - \frac{\varepsilon}{2}, \delta\beta - \frac{\varepsilon}{2})$-approximate solution to SMFR, where $\delta \in (0, 1 - 1/e]$ is the approximation factor of the approximation algorithm for SMK or SMM. If there is no set $S \in \mathcal{I}$ with $F'_{\alpha,\beta}(S) = d + 1$, then no (α, β)-approximate solution to SMFR exists.*

Proof. See Appendix A.1 for the proof. □

Algorithm 1: SMFR-Saturate

```
Input: Normalized, monotone, and submodular set functions $f, g_1, \ldots, g_d : 2^V \to \mathbb{R}^+$; Cost function $c : V \to \mathbb{R}^+$ and budget $k \in \mathbb{R}^+$ (for knapsack constraint) or Collection of feasible sets $\mathcal{I}(\mathcal{M}) \subseteq 2^V$ and rank $r \in \mathbb{Z}^+$ (for matroid constraint); Error parameter $\varepsilon \in (0, 1)$
Result: A set $\mathcal{S}$ of approximate solutions to SMFR
Initialize $\mathcal{S} \leftarrow \emptyset$;
Run an algorithm for SMK or SMM to maximize $f, g_1, \ldots, g_d$ subject to the constraint $\mathcal{I}_k$ or $\mathcal{I}(\mathcal{M})$ to compute $\mathtt{OPT}'_f, \mathtt{OPT}'_{g_1}, \ldots, \mathtt{OPT}'_{g_d}$;
for $\beta \leftarrow 0$; $\beta \leq 1$; $\beta \leftarrow \beta + \frac{\varepsilon}{2}$ do
    Initialize $\alpha_{max} \leftarrow 1$, $\alpha_{min} \leftarrow 0$;
    while $\alpha_{max} - \alpha_{min} > \frac{\varepsilon}{2}$ do
        Set $\alpha \leftarrow (\alpha_{max} + \alpha_{min})/2$ and define $F'_{\alpha,\beta}(S)$ according to Eq. 3;
        $S \leftarrow$ CostEffectiveGreedy($f, g_1, \ldots, g_d$, $c$, $k$, $\varepsilon$) (for knapsack constraint) or IterativeGreedy($f, g_1, \ldots, g_d$, $\mathcal{I}(\mathcal{M})$, $\varepsilon$) (for matroid constraint);
        if $F'_{\alpha,\beta}(S) \geq d + 1 - \frac{\varepsilon}{2}$ then
            $\alpha_{min} \leftarrow \alpha$ and $S_{\alpha,\beta} \leftarrow S$;
        else
            $\alpha_{max} \leftarrow \alpha$;
        end
    end
    Add $S_{\alpha_{min},\beta}$ to $\mathcal{S}$ and remove all $S_{\alpha',\beta'}$ with $\alpha' \leq \alpha_{min}$ and $\beta' < \beta$ from $\mathcal{S}$;
end
return $\mathcal{S}$;

Function CostEffectiveGreedy($f, g_1, \ldots, g_d$, $c$, $k$, $\varepsilon$):
    Initialize $S \leftarrow \emptyset$;
    while $\exists v \in V \setminus S$ such that $c(S \cup \{v\}) \leq k(1 + \ln \frac{2d+2}{\varepsilon})$ do
        $I \leftarrow \{v \in V : c(S \cup \{v\}) \leq k(1 + \ln \frac{2d+2}{\varepsilon})\}$;
        $v^* \leftarrow \arg\max_{v \in I} \big(F'_{\alpha,\beta}(S \cup \{v\}) - F'_{\alpha,\beta}(S)\big)/c(v)$ and $S \leftarrow S \cup \{v^*\}$;
    end
    return $S$;

Function IterativeGreedy($f, g_1, \ldots, g_d$, $\mathcal{I}(\mathcal{M})$, $\varepsilon$):
    for $l \leftarrow 1$; $l \leq 1 + \lceil \log_2 \frac{d+1}{\varepsilon} \rceil$; $l \leftarrow l + 1$ do
        $S_l \leftarrow \emptyset$;
        while $\exists v \in V : S_l \cup \{v\} \in \mathcal{I}(\mathcal{M})$ do
            $I \leftarrow \{v \in V : S_l \cup \{v\} \in \mathcal{I}(\mathcal{M})\}$;
            $v^* \leftarrow \arg\max_{v \in I} F'_{\alpha,\beta}(\cup_{j=1}^{l} S_j \cup \{v\}) - F'_{\alpha,\beta}(\cup_{j=1}^{l} S_j)$ and $S_l \leftarrow S_l \cup \{v^*\}$;
        end
    end
    return $S \leftarrow \bigcup_{l=1}^{1+\lceil \log_2 \frac{d+1}{\varepsilon} \rceil} S_l$;
```

Based on Lemma 1, we propose SMFR-Saturate in Algorithm 1 for SMFR. Generally, SMFR-Saturate follows the same framework to handle the knapsack and matroid constraints but uses different greedy algorithms to obtain approximate solutions to SC on $F'_{\alpha,\beta}$. We first run an algorithm for SMK or SMM on each objective function individually with the same knapsack constraint $\mathcal{I}_k$ or matroid constraint $\mathcal{I}(\mathcal{M})$ to calculate $\mathtt{OPT}'_f, \mathtt{OPT}'_{g_1}, \ldots, \mathtt{OPT}'_{g_d}$. Then, we iterate over each value of β from 0 to 1 with an interval of $\frac{\varepsilon}{2}$. For each value of β, we perform a bisection search on α between 0 and 1. Given a pair of α and β, we formulate an instance of SC on $F'_{\alpha,\beta}$ in Eq. 3.

To address SC on $F'_{\alpha,\beta}$, we adopt two different types of greedy algorithms specific to the knapsack and matroid constraints, respectively. For a knapsack constraint $\mathcal{I}_k$, we run the `CostEffectiveGreedy` algorithm, which starts from $S = \emptyset$ and adds the most "cost-effective" item v^* with the largest ratio between its marginal gain w.r.t. S and its cost $c(v^*)$ until no more items can be added with a relaxed knapsack constraint with a budget $k(1 + \ln \frac{2d+2}{\varepsilon})$, to find the candidate solution S. For a matroid constraint $\mathcal{I}(\mathcal{M})$, we run the `IterativeGreedy` algorithm, which performs the classic greedy algorithm for SMM (Fisher et al., 1978) iteratively in $1 + \lceil \log_2 \frac{d+1}{\varepsilon} \rceil$ rounds. In the l-th round, we start from $S_l = \emptyset$ and add the item v^* that satisfies $S_l \cup \{v^*\} \in \mathcal{I}(\mathcal{M})$ and has the largest marginal gain w.r.t. $\cup_{j=1}^{l} S_j$ until no more items can be added to S_l under the knapsack constraint $\mathcal{I}(\mathcal{M})$. Finally, we return the union of the items selected over all rounds, i.e., $\bigcup_{l=1}^{1+\lceil \log_2 \frac{d+1}{\varepsilon} \rceil} S_l$, as the candidate solution S.

After computing a candidate solution S, if $F'_{\alpha,\beta}(S) \geq d + 1 - \frac{\varepsilon}{2}$, that is, S reaches the "saturation level" w.r.t. α, β according to Lemma 1, we set S as the current solution $S_{\alpha,\beta}$ and search in the upper half for a better solution with a higher value of α; otherwise, we search in the lower half for a feasible solution. When $\alpha_{max} - \alpha_{min} \leq \frac{\varepsilon}{2}$, we add the solution $S_{\alpha_{min},\beta}$ to $\mathcal{S}$, remove all solutions dominated by $S_{\alpha_{min},\beta}$, and move on to the next value of β. Finally, all non-dominated solutions in $\mathcal{S}$ are returned for SMFR.

The theoretical guarantees of SMFR-Saturate for SMFR with knapsack and matroid constraints are analyzed in the following two theorems, respectively.

Theorem 1. *For SMFR with a knapsack constraint $\mathcal{I}_k$, SMFR-Saturate runs in $O(dt(\mathcal{A}) + \frac{n^2}{\varepsilon} \log \frac{1}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the δ-approximation algorithm for SMK, and provides a set $\mathcal{S}$ of solutions with the following properties: (1) $|\mathcal{S}| = O(\frac{1}{\varepsilon})$, (2) $c(S) = O(k \log \frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$, (3) for each (α^*, β^*)-approximate Pareto optimal solution S^* to SMFR, there must exist its corresponding solution $S \in \mathcal{S}$ such that $f(S) \geq (\delta\alpha^* - \varepsilon)\mathtt{OPT}_f$ and $g_i(S) \geq (\delta\beta^* - \varepsilon)\mathtt{OPT}_{g_i}, \forall i \in [d]$.*

Proof. See Appendix A.2 for the proof. ☐

Theorem 2. *For SMFR with a matroid constraint $\mathcal{I}(\mathcal{M})$, SMFR-Saturate runs in $O(dt(\mathcal{A}) + \frac{nr}{\varepsilon} \log^2 \frac{d}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the δ-approximation algorithm for SMM, and provides a set $\mathcal{S}$ of solutions with the following properties: (1) $|\mathcal{S}| = O(\frac{1}{\varepsilon})$, (2) $|S| = O(r \log \frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$, (3) for each (α^*, β^*)-approximate Pareto optimal solution S^* to SMFR, there must exist its corresponding solution $S \in \mathcal{S}$ such that $f(S) \geq (\delta\alpha^* - \varepsilon)\mathtt{OPT}_f$ and $g_i(S) \geq (\delta\beta^* - \varepsilon)\mathtt{OPT}_{g_i}, \forall i \in [d]$.*

Proof. See Appendix A.3 for the proof. ☐

5 Experiments

In this section, we present extensive experimental results to evaluate the performance of our proposed algorithm (SMFR-Saturate) on two benchmark problems, namely *Maximum Coverage* and *Recommendation*, using several real-world datasets. We compare SMFR-Saturate with the following non-trivial baselines.

- Greedy+Max (or Greedy): The original greedy algorithms for single-objective submodular maximization. For SMK, we adopt the $O(n^2)$-time Greedy+Max algorithm by Yaroslavtsev et al. (2020); and for SMM, we adopt the $O(nr)$-time Greedy algorithm by Fisher et al. (1978). Both algorithms have the same approximation factor of 1/2.
- Saturate: The bicriteria approximation algorithms for the problem of *multi-objective submodular maximization* (MOSM) that maximizes the minimum among multiple (submodular) objective functions. As for SMFR, we should maximize the minimum among the $d + 1$ functions of f and $g_1, \ldots, g_d$. In particular, Saturate for MOSM with knapsack and matroid constraints is presented in (Krause et al., 2008) and (Anari et al., 2019), respectively.
- SMSC: A $(0.16, 0.16)$-approximation algorithm for the problem of Submodular Maximization under Submodular Cover (SMSC) (Ohsaka & Matsuoka, 2021), which can be used for SMFR only when $d = 1$ by maximizing f under the submodular cover constraint defined on g_1.

- BSM-Saturate: The instance-dependent bicriteria approximation algorithm for balancing *utility* (i.e., maximizing f) and *fairness* (i.e., maximizing the minimum of $g_1, \ldots, g_d$) in (Wang et al., 2024).
- OPT: Formulating an instance of SMFR as an integer-linear program (ILP) and using a solver to enumerate its Pareto optimal solutions in the worst-case exponential time. The ILP formulations of SMFR for *Maximum Coverage* and *Recommendation* are deferred to Appendix B.

All algorithms are appropriately adapted to provide solutions without violating the specified constraints. We implemented them in Python 3, and for the OPT algorithm, we applied the Gurobi[2] optimizer to solve the ILP formulations of the *Maximum Coverage* and *Recommendation* instances. All algorithms except OPT were accelerated using the lazy-forward strategy (Leskovec et al., 2007), as this strategy cannot be applied to OPT. All experiments were run on a MacBook Pro laptop with an Apple M1 Max processor and 32GB memory running MacOS 14. For reproducibility sake, our code and data have been published anonymously[3].

5.1 Maximum Coverage

Setup. In this subsection, we evaluate the performance of different algorithms for SMFR on the *Maximum Coverage* problem using two real-world datasets: *Facebook* and *DBLP*. The *Facebook* dataset (Mislove et al., 2010) is an undirected graph of $1,216$ nodes and $42,443$ edges representing the friendships between Rice University students on Facebook, and the *DBLP* dataset (Dong et al., 2023) is an undirected graph of $3,980$ nodes and $6,966$ edges denoting the coauthorships between researchers.

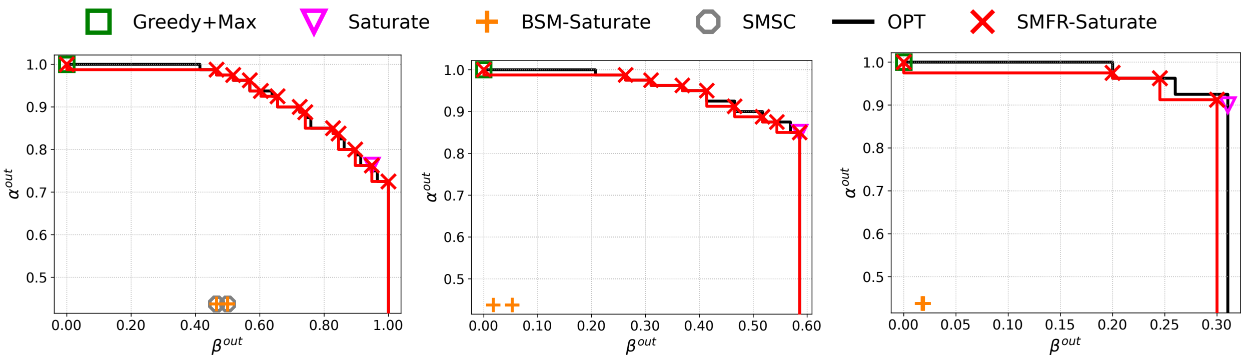
Our settings for *Maximum Coverage* follow those used in the existing literature on submodular maximization (Halabi et al., 2020; Ohsaka & Matsuoka, 2021; Wang et al., 2024). Given a graph $\mathcal{G} = (V, E)$, the utility (i.e., coverage) function is defined as $f(S) := |\bigcup_{v \in S} \mathcal{N}(v)|$, where $\mathcal{N}(v)$ is the set of nodes consisting of v and its neighbors in $\mathcal{G}$. That is, the coverage of a set $S \subseteq V$ is measured by the number of nodes in the union of the neighborhoods of all nodes in S. To define the representativeness functions $g_1, g_2, \ldots, g_d$, we divide the node set into d communities $C_1, \ldots, C_d$ such that $\bigcup_{i=1}^{d} C_i = V$. For each $i \in [d]$, the function g_i is associated with a particular community C_i as $g_i(S) := |\bigcup_{v \in S} \mathcal{N}(v) \cap C_i|$. That is, the representativeness of a set S for a community C_i is measured by the number of nodes in C_i covered by S. For both datasets, the node set V is partitioned into four disjoint groups using the Louvain method (Blondel et al., 2008) for community detection. We then index the four communities according to their sizes as $|C_1| \geq |C_2| \geq |C_3| \geq |C_4|$. For the *DBLP* dataset, we follow the scheme of (Jin et al., 2021) to define a knapsack constraint by assigning a cost of 0.2 times its degree to each node and then normalizing all costs by the minimum cost. For the *Facebook* dataset, we define a partition matroid constraint by dividing all nodes into 4 disjoint groups based on a sensitive attribute (i.e., *age*). We then follow the rule of *equal representation* (Halabi et al., 2020) to set the same upper bound $k \in \mathbb{Z}^+$ for each age group, resulting in a partition matroid of rank $r = 4k$.

Results. Figures 1a–1c and 2a–2c present the trade-offs between α and β achieved by each algorithm for different instances of SMFR on *Maximum Coverage* with knapsack and matroid constraints on the *DBLP* and *Facebook* datasets, respectively. We fix $k = 40$ for the knapsack constraint and $k = 5$ (and thus $r = 20$) for the matroid constraint. We set $d = 1, 2$, and 4 by considering the representativeness functions on the first group C_1, the first two groups C_1 and C_2, and all four groups from C_1 to C_4. In each of these figures, the x- and y-axes represent the values of α and β for all solutions with a distinct marker for each algorithm. Furthermore, we also use a black line and a red line to denote the optimal Pareto frontier returned by OPT and its approximation returned by SMFR-Saturate. From the results, we observe that the Pareto frontiers provided by SMFR-Saturate are equal or very close to the optimal ones. This confirms the effectiveness of SMFR-Saturate for the SMFR problem. We also find that the Greedy+Max and Greedy algorithms, which focus solely on maximizing f, generally provide solutions with low values of β, indicating a significant neglect of representativeness functions. Furthermore, Saturate, which maximizes the minimum among all representativeness and utility functions and does not allow for any trade-off between f and g by design, in some cases (e.g., Figures 1a–1c), it provides a solution with the highest β value while having a value of α equal or close to that of SMFR-Saturate and OPT for maximum β. However, it returns inferior

---

[2] `https://www.gurobi.com/solutions/gurobi-optimizer/`

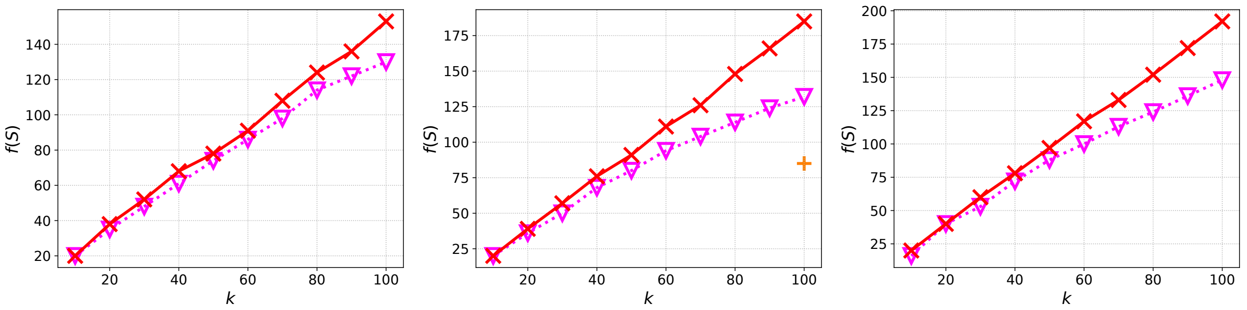
[3] `https://anonymous.4open.science/r/Fair-Representation-in-Submodular-Subset-Selection-A-Pareto-Optimization-Approach-B5B5`



(a) *DBLP* ($d = 1, k = 40$)

(b) *DBLP* ($d = 2, k = 40$)

(c) *DBLP* ($d = 4, k = 40$)



(d) *DBLP* ($d = 1, \beta \geq 0.8$)

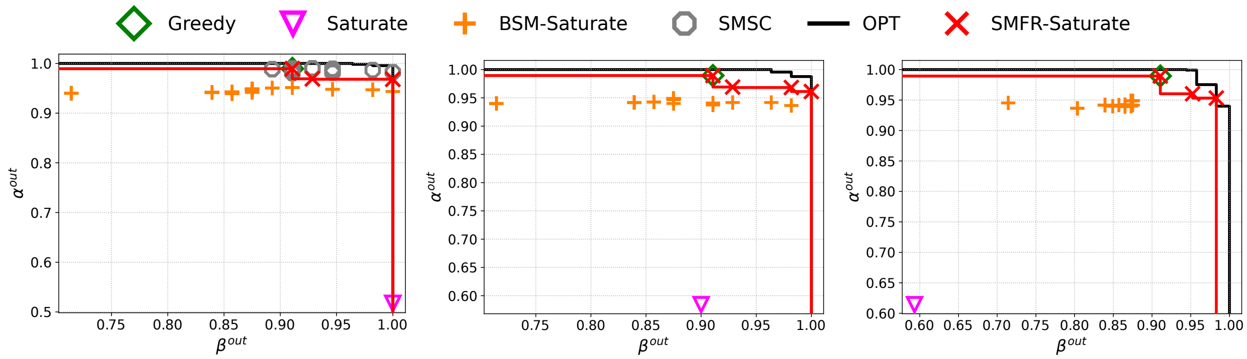
(e) *DBLP* ($d = 2, \beta \geq 0.4$)

(f) *DBLP* ($d = 4, \beta \geq 0.2$)

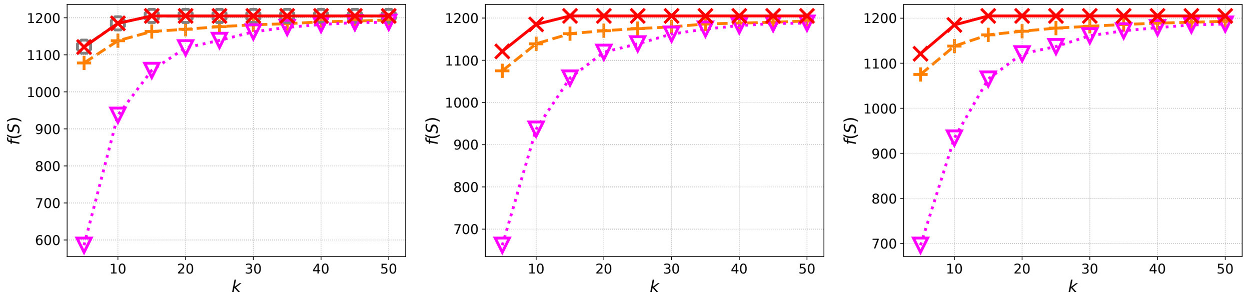
Figure 1: Results for *Maximum Coverage* on the *DBLP* dataset, with knapsack constraints.

solutions dominated by those of SMFR-SATURATE in other cases. BSM-SATURATE and SMSC provide different trade-offs between f and g by adjusting the threshold value τ in their definitions. The trade-offs reported by SMSC are marginally better than those of SMFR-SATURATE on the *Facebook* dataset with matroid constraints (Figure 2a). Conversely, it performs poorly for knapsack constraints (Figure 1a). In fact, SMSC is a special case of SMFR when $d = 1$, the matroid/knapsack constraint is reduced to the cardinality constraint, and the trade-off between f and g is predetermined by τ. It is also noted that SMSC cannot work when $d > 1$. Although BSM-SATURATE does not have the restriction of $d = 1$, its trade-offs are never better than those obtained by SMFR-SATURATE, and significantly worse for *Maximum Coverage* on the *DBLP* dataset with knapsack constraints (Figures 1a–1c).

Figures 1d–1f and 2d–2f report the effect of the parameter k, which directly decides the solution size, on the performance of each algorithm for different instances of SMFR in the context of *Maximum Coverage* with knapsack and matroid constraints on the *DBLP* and *Facebook* datasets, respectively. In each plot, the x-axis represents the value of k in the knapsack or matroid constraint, and the y-axis represents the maximum utility value $f(S)$ among all solutions with a certain level of representativeness, i.e., the value of β reaches a given threshold, provided by an algorithm. We also set $d = 1, 2$, and 4 by considering the representativeness functions on C_1, $C_1\&C_2$, and C_1–C_4. Only solutions with $\beta \geq 0.8$ are considered for $d = 1$, $\beta \geq 0.4$ for $d = 2$, and $\beta \geq 0.2$ for $d = 4$. A unique marker and a distinct line color are used for each algorithm. From Figures 1d–1f, we observe that the solutions provided by SMFR-SATURATE consistently achieve the highest utility value $f(S)$ across all values of k in the knapsack constraint. The absence of SMSC and BSM-SATURATE indicates that they fail to provide solutions with an adequate level of representativeness (i.e., the value of β is below the given thresholds), with the only exception shown in Figure 1e when $k = 100$. Furthermore, although SATURATE provides valid solutions in all cases, the gap in the utility value $f(S)$ between SMFR-SATURATE and SATURATE widens as the knapsack restriction becomes less stringent (i.e., increasing k), for all values of the number of representativeness functions d. Figures 2d–2f show that across all values of k, the solutions provided by SMFR-SATURATE always achieve utility values $f(S)$ higher than those of BSM-SATURATE and SATURATE. Unlike the case of knapsack constraints, the gap in the utility



(a) *Facebook* $(d = 1, k = 5)$ (b) *Facebook* $(d = 2, k = 5)$ (c) *Facebook* $(d = 4, k = 5)$



(d) *Facebook* $(d = 1, \beta \geq 0.8)$ (e) *Facebook* $(d = 2, \beta \geq 0.4)$ (f) *Facebook* $(d = 4, \beta \geq 0.2)$

Figure 2: Results for *Maximum Coverage* on the *Facebook* dataset, with matroid constraints.

value $f(S)$ among all methods decreases as the matroid constraint becomes less stringent (i.e. increasing k), for all values of the number of representativeness functions d. In the case of $d = 1$, SMSC and SMFR-Saturate exhibit the same performance, as shown in Figure 2d. The above results confirm that when the trade-off level between f and g is pre-specified, one can still find a corresponding solution from those of SMFR-Saturate that is comparable to or better than those provided by other baselines.

5.2 Recommendation

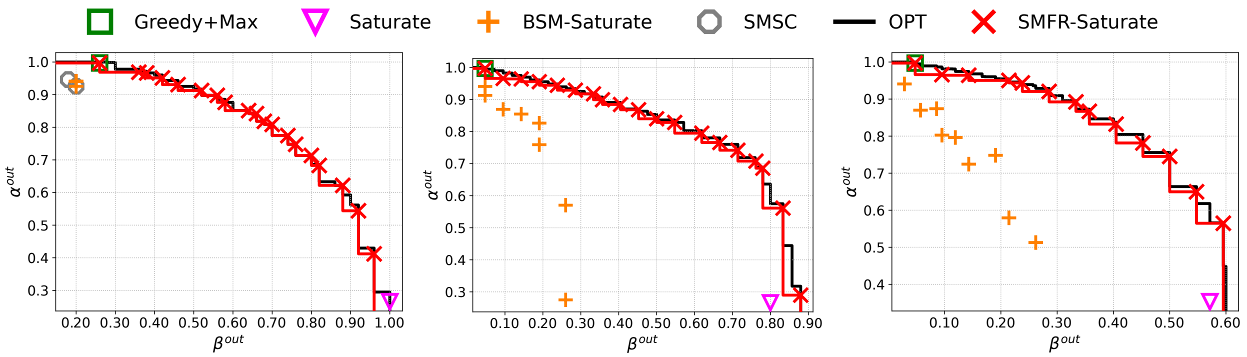
Setup. In this subsection, we evaluate the performance of different algorithms for SMFR on the *Recommendation* problem using another two real-world datasets: *X-Wines* (de Azambuja et al., 2023) and *MovieLens*[4]. The *X-Wines* dataset consists of 150 000 ratings from 10 561 users on 1 007 wines, where each rating takes a value in the range $[1.0, 1.5, \ldots, 5.0]$. Moreover, each wine in the dataset is associated with one or more food types that pair with the wine itself; we group these food types into four categories: *"meat"*, *"fish"*, *"pasta"*, and *"cheese"*. The *MovieLens* dataset consists of 100 000 ratings from 600 users on 9 000 movies, where each rating takes a value in the range $[0.5, 1.0, \ldots, 5.0]$. Each movie in the dataset is associated with one or more genres, with a total of 20 genres.

Our experimental settings are similar to those adopted in (Ohsaka & Matsuoka, 2021). In the following, we use the term *"item"* to refer to either a wine in the *X-Wines* dataset or a movie in the *MovieLens* dataset. By performing the non-negative matrix factorization[5] (NMF) on the user-item rating matrix with $p = 32$ factors, we obtain a 32-dimensional feature vector for each item and user. Denoting by $\mathbf{v}_i \in \mathbb{R}^p$ the feature vector of item i, and by $\mathbf{u}_j \in \mathbb{R}^p$ the feature vector of user j, the inner product $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ between two feature vectors associated with two items measures their similarity. The same holds for users and items as well: $\langle \mathbf{v}_i, \mathbf{u}_j \rangle$ indicates the level at which a user likes an item. To design the utility function f according to the facility location objective, we select a subset T of items with at least 54 ratings ($|T| = 503$ for the *X-Wines* dataset,

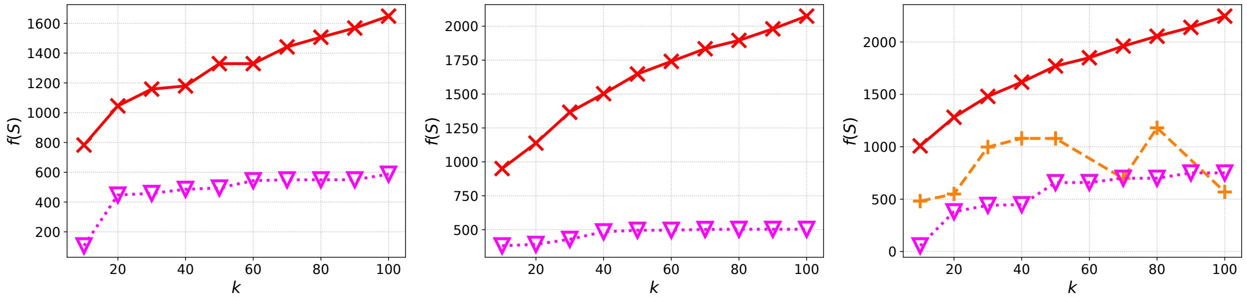
---

[4] `https://grouplens.org/datasets/movielens/`

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html`



(a) *X-Wines* $(d = 1, k = 50)$ (b) *X-Wines* $(d = 2, k = 50)$ (c) *X-Wines* $(d = 4, k = 50)$

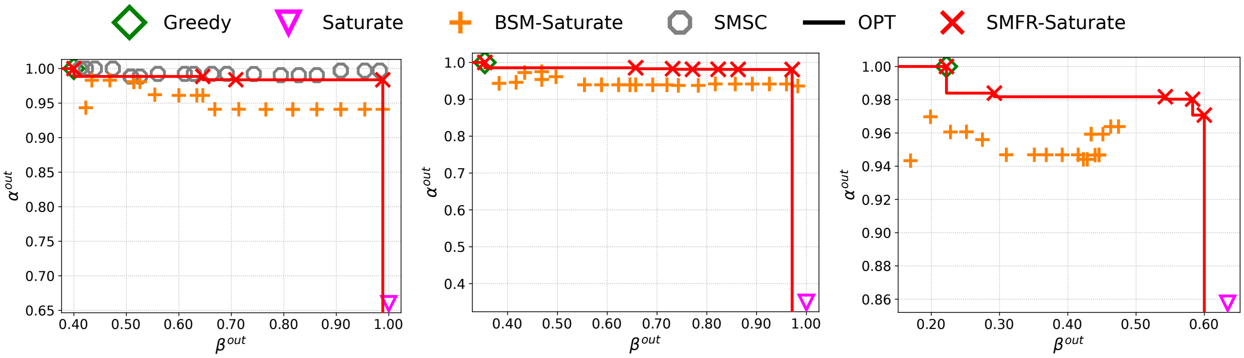


(d) *X-Wines* $(d = 1, \beta \geq 0.8)$ (e) *X-Wines* $(d = 2, \beta \geq 0.4)$ (f) *X-Wines* $(d = 4, \beta \geq 0.2)$

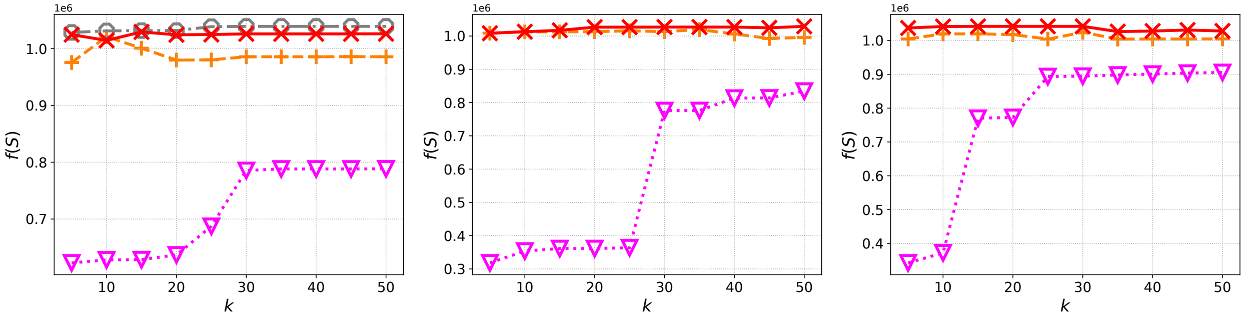
Figure 3: Results for *Recommendation* on the *X-Wines* dataset, with knapsack constraints.

and $|T| = 403$ for the *MovieLens* dataset), and define $f : 2^V \to \mathbb{R}^+$ as $f(S) := \sum_{t \in T} \max_{s \in S} \langle \mathbf{v}_s, \mathbf{v}_t \rangle$, where V is the set of all items in each dataset: $|V| = 1\,007$ for the *X-Wines* dataset, and $|V| = 9\,000$ for the *MovieLens* dataset. The function f captures how well the selected subset S can represent all items in T in the sense that for any item $t \in T$, there exists an item in S that is highly similar to it. This function, as defined, is known to be monotone and submodular (Frieze, 1974). To define the representativeness functions $g_1, g_2, \ldots, g_d$, we consider using, for the *X-Wines* dataset the food type categories with which a wine pair, and for the *MovieLens* dataset the genres to which a movie belongs. Specifically, for the *X-Wines* dataset, we divide wines into four groups according to their associated food type categories as G_1 (*meat*), G_2 (*fish*), G_3 (*pasta*), and G_4 (*cheese*). Similarly, for the *MovieLens* dataset, we divide movies into four groups according to their genres as G_1 (*dramas*), G_2 (*comedies*), G_3 (*thrillers*), and G_4 (*action movies*). Then, each g_i function is associated with a particular set of items and is defined as $g_i(S) := |S \cap G_i|$. To be specific, the representativeness of S for G_i is measured by the number of items in S selected from G_i. For the *X-Wines* dataset, we define a knapsack constraint by assigning to each item (wine) a random integer cost in the range $[1, 10]$. For the *MovieLens* dataset, to define a matroid constraint, we partition the movies into 7 groups according to their release dates: $[1900, 1950)$, $[1950, 1970)$, $[1970, 1980)$, $[1980, 1990)$, $[1990, 2000)$, $[2000, 2010)$, and $[2010, 2019)$. We also use an equal upper bound $k \in \mathbb{Z}^+$ for each group, resulting in a partition matroid of rank $r = 7k$.

Results. Figures 3 and 4 present the performance of each algorithm for different instances of SMFR on *Recommendation* with knapsack and matroid constraints on the *X-Wines* and *MovieLens* datasets, respectively. In general, we observe results similar to those for *Maximum Coverage* and further confirm the effectiveness of SMFR-Saturate for SMFR in different applications. The absence of OPT in Figures 4a–4c is due to the inefficiency of the ILP solver: it cannot finish on any SMFR instance for the *MovieLens* dataset within one hour. We also find that SMFR-Saturate shows more significant advantages over SMSC and BSM-Saturate for the knapsack constraints than for the matroid constraints. In particular, SMSC slightly outperforms SMFR-Saturate when $d = 1$ on the *MovieLens* dataset, with matroid constraints. This is because the solutions with cardinality constraints are typically very close to those with the partition



(a) *MovieLens* ($d = 1, k = 25$)
(b) *MovieLens* ($d = 2, k = 25$)
(c) *MovieLens* ($d = 4, k = 25$)



(d) *MovieLens* ($d = 1, \beta \geq 0.8$)
(e) *MovieLens* ($d = 2, \beta \geq 0.4$)
(f) *MovieLens* ($d = 4, \beta \geq 0.2$)

Figure 4: Results for *Recommendation* on the *MovieLens* dataset, with matroid constraints.

matroid constraints that we define but differ significantly from those with knapsack constraints. As such, SMSC, which is designed specifically for cardinality constraints, achieves good performance under matroid constraints without adaptations. Again, we note that SMSC is not comparable to SMFR-Saturate in other cases.

Finally, we omit the remaining experimental results due to space limitations. Please refer to Appendix C for those results, which further confirm the effectiveness of SMFR-Saturate in other experimental settings and provide additional evaluations for the efficiency of SMFR-Saturate and other baselines.

6 Conclusion and Future Work

In this paper, we study a novel multi-objective combinatorial optimization problem called *Submodular Maximization with Fair Representation* (SMFR), which aims to select subsets from a ground set under a specific knapsack or matroid constraint such that a submodular (utility) function f is maximized while d submodular (representativeness) functions $g_1, \ldots, g_d$ are also maximized. We show the hardness of finding optimal solutions to SMFR and propose a Pareto optimization approach, SMFR-Saturate, to enumerating a set of approximate solutions to all Pareto optimal solutions with different trade-offs between multiple objectives for SMFR. Finally, we demonstrate the effectiveness of SMFR-Saturate in two classic submodular problems, *Maximum Coverage* and *Recommendation*, using real-world data.

In future work, we would like to extend SMFR to more general classes of functions in subset selection problems, including non-monotone and weakly submodular functions. In addition, it would also be interesting to expand the realm of *fair submodular optimization* (Halabi et al., 2023; Mehrotra & Vishnoi, 2023) by considering more novel and practical notions of fairness.

References

Nima Anari, Nika Haghtalab, Seffi Naor, Sebastian Pokutta, Mohit Singh, and Alfredo Torrico. Structured robust submodular maximization: Offline and online algorithms. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 3128–3137. PMLR, 2019. URL `http://proceedings.mlr.press/v89/anari19a.html`.

Çigdem Aslay, Wei Lu, Francesco Bonchi, Amit Goyal, and Laks V. S. Lakshmanan. Viral marketing meets social advertising: Ad allocation with minimum regret. *Proc. VLDB Endow.*, 8(7):822–833, 2015. URL `http://www.vldb.org/pvldb/vol8/p814-aslay.pdf`.

Çigdem Aslay, Francesco Bonchi, Laks V. S. Lakshmanan, and Wei Lu. Revenue maximization in incentivized social advertising. *Proc. VLDB Endow.*, 10(11):1238–1249, 2017. URL `http://www.vldb.org/pvldb/vol10/p1238-aslay.pdf`.

Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *Proceedings of the 2014 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1497–1514. Society for Industrial and Applied Mathematics, 2013. URL `https://doi.org/10.1137/1.9781611973402.110`.

Wei-Xuan Bao, Jun-Yi Hang, and Min-Ling Zhang. Submodular feature selection for partial label learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*, pp. 26–34. Association for Computing Machinery, 2022. URL `https://doi.org/10.1145/3534678.3539292`.

Ruben Becker, Federico Corò, Gianlorenzo D'Angelo, and Hugo Gilbert. Balancing spreads of influence in a social network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):3–10, 2020. URL `https://doi.org/10.1609/aaai.v34i01.5327`.

Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech.: Theory Exp.*, 2008(10):P10008, 2008. URL `https://dx.doi.org/10.1088/1742-5468/2008/10/P10008`.

Niv Buchbinder, Moran Feldman, and Mohit Garg. Deterministic $(1/2 + \varepsilon)$-approximation for submodular maximization over a matroid. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 241–254. SIAM, 2019. URL `https://doi.org/10.1137/1.9781611975482.16`.

Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011. URL `https://doi.org/10.1137/080733991`.

Rogério Xavier de Azambuja, A. Jorge Morais, and Vítor Filipe. X-Wines: A wine dataset for recommender systems and machine learning. *Big Data Cogn. Comput.*, 7(1):20, 2023. URL `https://doi.org/10.3390/bdcc7010020`.

Yushun Dong, Jing Ma, Song Wang, Chen Chen, and Jundong Li. Fairness in graph mining: A survey. *IEEE Trans. Knowl. Data Eng.*, 35(10):10583–10602, 2023. URL `https://doi.org/10.1109/TKDE.2023.3265598`.

Alina Ene and Huy L. Nguyen. A nearly-linear time algorithm for submodular maximization with a knapsack constraint. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, pp. 53:1–53:12, Dagstuhl, Germany, 2019a. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL `https://doi.org/10.4230/LIPIcs.ICALP.2019.53`.

Alina Ene and Huy L. Nguyen. Towards nearly-linear time algorithms for submodular maximization with a matroid constraint. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, pp. 54:1–54:14, Dagstuhl, Germany, 2019b. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL `https://doi.org/10.4230/LIPIcs.ICALP.2019.54`.

Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998. URL `https://doi.org/10.1145/285055.285059`.

Moran Feldman, Zeev Nutov, and Elad Shoham. Practical budgeted submodular maximization. *Algorithmica*, 85(5):1332–1371, 2022. URL `https://doi.org/10.1007/s00453-022-01071-2`.

Chao Feng and Chao Qian. Multi-objective submodular maximization by regret ratio minimization with theoretical guarantee. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12302–12310, 2021. URL `https://doi.org/10.1609/aaai.v35i14.17460`.

Yuval Filmus and Justin Ward. Monotone submodular maximization over a matroid via non-oblivious local search. *SIAM J. Comput.*, 43(2):514–542, 2014. URL `https://doi.org/10.1137/130920277`.

Marshall L. Fisher, George L. Nemhauser, and Laurence A. Wolsey. An analysis of approximations for maximizing submodular set functions–II. In Michel L. Balinski and Alan J. Hoffman (eds.), *Polyhedral Combinatorics: Dedicated to the memory of D.R. Fulkerson*, pp. 73–87. Springer, Berlin, Heidelberg, 1978. URL `https://doi.org/10.1007/BFb0121195`.

Alan M. Frieze. A cost function property for plant location problems. *Math. Program.*, 7(1):245–248, 1974. URL `https://doi.org/10.1007/BF01585521`.

Marwa El Halabi, Slobodan Mitrovic, Ashkan Norouzi-Fard, Jakab Tardos, and Jakub Tarnawski. Fairness in streaming submodular maximization: Algorithms and hardness. *Advances in Neural Information Processing Systems*, 33:13609–13622, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/9d752cb08ef466fc480fba981cfa44a1-Abstract.html`.

Marwa El Halabi, Federico Fusco, Ashkan Norouzi-Fard, Jakab Tardos, and Jakub Tarnawski. Fairness in streaming submodular maximization over a matroid constraint. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 9150–9171. PMLR, 2023. URL `https://proceedings.mlr.press/v202/el-halabi23a.html`.

Kai Han, Shuang Cui, Tianshuai Zhu, Enpei Zhang, Benwei Wu, Zhizhuo Yin, Tong Xu, Shaojie Tang, and He Huang. Approximation algorithms for submodular data summarization with a knapsack constraint. *Proc. ACM Meas. Anal. Comput. Syst.*, 5(1):05:1–05:31, 2021. URL `https://doi.org/10.1145/3447383`.

Chien-Chung Huang, Naonori Kakimura, and Yuichi Yoshida. Streaming algorithms for maximizing monotone submodular functions under a knapsack constraint. *Algorithmica*, 82(4):1006–1032, 2020. URL `https://doi.org/10.1007/s00453-020-00786-4`.

Tianyuan Jin, Yu Yang, Renchi Yang, Jieming Shi, Keke Huang, and Xiaokui Xiao. Unconstrained submodular maximization with modular costs: Tight approximation and application to profit maximization. *Proc. VLDB Endow.*, 14(10):1756–1768, 2021. URL `http://www.vldb.org/pvldb/vol14/p1756-jin.pdf`.

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)*, pp. 137–146. Association for Computing Machinery, 2003. URL `https://doi.org/10.1145/956750.956769`.

Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. The budgeted maximum coverage problem. *Inf. Process. Lett.*, 70(1):39–45, 1999. URL `https://doi.org/10.1016/S0020-0190(99)00031-9`.

Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, pp. 71–104. Cambridge University Press, Cambridge, UK, 2014. URL `https://doi.org/10.1017/CBO9781139177801.004`.

Andreas Krause and Carlos Guestrin. A note on the budgeted maximization of submodular functions. Technical Report CMU-CALD-05-103, Carnegie Mellon University, 2005. URL `https://las.inf.ethz.ch/files/krause05note.pdf`.

Andreas Krause, H. Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *J. Mach. Learn. Res.*, 9(93):2761–2801, 2008. URL `https://jmlr.org/papers/v9/krause08b.html`.

Ariel Kulik, Roy Schwartz, and Hadas Shachnai. A refined analysis of submodular greedy. *Oper. Res. Lett.*, 49(4):507–514, 2021. URL `https://doi.org/10.1016/j.orl.2021.04.006`.

Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*, pp. 420–429. Association for Computing Machinery, 2007. URL `https://doi.org/10.1145/1281192.1281239`.

Wenxin Li, Moran Feldman, Ehsan Kazemi, and Amin Karbasi. Submodular maximization in clean linear time. *Advances in Neural Information Processing Systems*, 35:17473–17487, 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/hash/6faf3b8ed0df532c14d0fc009e451b6d-Abstract-Conference.html`.

Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 912–920. Association for Computational Linguistics, 2010. URL `https://aclanthology.org/N10-1134/`.

Yuzong Liu, Kai Wei, Katrin Kirchhoff, Yisong Song, and Jeff A. Bilmes. Submodular feature selection for high-dimensional acoustic score spaces. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7184–7188. IEEE, 2013. URL `https://doi.org/10.1109/ICASSP.2013.6639057`.

Anay Mehrotra and Nisheeth K. Vishnoi. Maximizing submodular functions for recommendation in the presence of biases. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*, pp. 3625–3636. Association for Computing Machinery, 2023. URL `https://doi.org/10.1145/3543507.3583195`.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, and Amin Karbasi. Fast constrained submodular maximization: Personalized data summarization. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, pp. 1358–1367. PMLR, 2016. URL `http://proceedings.mlr.press/v48/mirzasoleiman16.html`.

Alan Mislove, Bimal Viswanath, Krishna P. Gummadi, and Peter Druschel. You are who you know: Inferring user profiles in online social networks. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM '10)*, pp. 251–260. Association for Computing Machinery, 2010. URL `https://doi.org/10.1145/1718487.1718519`.

George L. Nemhauser and Laurence A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, 3(3):177–188, 1978. URL `https://doi.org/10.1287/moor.3.3.177`.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Math. Program.*, 14:265–294, 1978. URL `https://doi.org/10.1007/BF01588971`.

Naoto Ohsaka and Tatsuya Matsuoka. Approximation algorithm for submodular maximization under submodular cover. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 792–801. PMLR, 2021. URL `https://proceedings.mlr.press/v161/ohsaka21a.html`.

Chao Qian, Yang Yu, and Zhi-Hua Zhou. Subset selection by pareto optimization. *Advances in Neural Information Processing Systems*, 28:1774–1782, 2015. URL `https://proceedings.neurips.cc/paper/2015/hash/b4d168b48157c623fbd095b4a565b5bb-Abstract.html`.

Chao Qian, Jing-Cheng Shi, Yang Yu, Ke Tang, and Zhi-Hua Zhou. Subset selection under noise. *Advances in Neural Information Processing Systems*, 30:3560–3570, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/d7a84628c025d30f7b2c52c958767e76-Abstract.html`.

Chao Qian, Chao Bian, and Chao Feng. Subset selection by pareto optimization with recombination. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2408–2415, 2020. URL `https://doi.org/10.1609/aaai.v34i03.5621`.

Vahid Roostapour, Aneta Neumann, Frank Neumann, and Tobias Friedrich. Pareto optimization for subset selection with dynamic cost constraints. *Artif. Intell.*, 302:103597, 2022. URL `https://doi.org/10.1016/j.artint.2021.103597`.

Tasuku Soma and Yuichi Yoshida. Regret ratio minimization in multi-objective submodular function maximization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1):905–911, 2017. URL `https://doi.org/10.1609/aaai.v31i1.10652`.

Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004. URL `https://doi.org/10.1016/S0167-6377(03)00062-2`.

Jing Tang, Xueyan Tang, Andrew Lim, Kai Han, Chongshou Li, and Junsong Yuan. Revisiting modified greedy algorithm for monotone submodular maximization with a knapsack constraint. *Proc. ACM Meas. Anal. Comput. Syst.*, 5(1):08:1–08:22, 2021. URL `https://doi.org/10.1145/3447386`.

Shaojie Tang. When social advertising meets viral marketing: Sequencing social advertisements for influence maximization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1):176–183, 2018. URL `https://doi.org/10.1609/aaai.v32i1.11306`.

Shaojie Tang and Jing Yuan. Beyond submodularity: a unified framework of randomized set selection with group fairness constraints. *J. Comb. Optim.*, 45(4):102, 2023. URL `https://doi.org/10.1007/s10878-023-01035-4`.

Alfredo Torrico, Mohit Singh, Sebastian Pokutta, Nika Haghtalab, Joseph (Seffi) Naor, and Nima Anari. Structured robust submodular maximization: Offline and online algorithms. *INFORMS J. Comput.*, 33 (4):1590–1607, 2021. URL `https://doi.org/10.1287/ijoc.2020.0998`.

Alan Tsang, Bryan Wilder, Eric Rice, Milind Tambe, and Yair Zick. Group-fairness in influence maximization. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 5997–6005. International Joint Conferences on Artificial Intelligence Organization, 2019. URL `https://doi.org/10.24963/ijcai.2019/831`.

Sebastian Tschiatschek, Adish Singla, and Andreas Krause. Selecting sequences of items via submodular maximization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1):2667–2673, 2017. URL `https://doi.org/10.1609/aaai.v31i1.10923`.

Rajan Udwani. Multi-objective maximization of monotone submodular functions with cardinality constraint. *Advances in Neural Information Processing Systems*, 31:9513–9524, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/7e448ed9dd44e6e22442dac8e21856ae-Abstract.html`.

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)*, pp. 67–74. Association for Computing Machinery, 2008. URL `https://doi.org/10.1145/1374376.1374389`.

Yanhao Wang, Jiping Zheng, and Fanxu Meng. Improved algorithm for regret ratio minimization in multi-objective submodular maximization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37 (10):12500–12508, 2023. URL `https://doi.org/10.1609/aaai.v37i10.26472`.

Yanhao Wang, Yuchen Li, Francesco Bonchi, and Ying Wang. Balancing utility and fairness in submodular maximization. In *Proceedings of the 27th International Conference on Extending Database Technology, EDBT 2024, Paestum, Italy, March 25 - March 28*, pp. 1–14. OpenProceedings.org, 2024. URL `https://doi.org/10.48786/edbt.2024.01`.

Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982. URL `https://doi.org/10.1007/BF02579435`.

Grigory Yaroslavtsev, Samson Zhou, and Dmitrii Avdiukhin. “Bring your own greedy”+max: Near-optimal 1/2-approximations for submodular knapsack. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 3263–3274. PMLR, 2020. URL `http://proceedings.mlr.press/v108/yaroslavtsev20a.html`.

A Proofs of Lemmas and Theorems

A.1 Proof of Lemma 1

Lemma 1. *If $F'_{\alpha,\beta}(S) \geq d+1-\frac{\varepsilon}{2}$ for any set $S \in \mathcal{I}$, then S is a $(\delta\alpha-\frac{\varepsilon}{2}, \delta\beta-\frac{\varepsilon}{2})$-approximate solution to SMFR, where $\delta \in (0, 1-1/e]$ is the approximation factor of the approximation algorithm for SMK or SMM. If there is no set $S \in \mathcal{I}$ with $F'_{\alpha,\beta}(S) = d+1$, then no (α,β)-approximate solution to SMFR exists.*

Proof. For the proof of the first statement, we first consider the two special cases of $\alpha = 0$ and $\beta = 0$. When $\alpha = 0$ or $\beta = 0$, if $F'_{\alpha,\beta}(S) > d+1-\frac{\varepsilon}{2}$, we will have $\frac{g_i(S)}{\beta \mathtt{OPT}'_{g_i}} > 1-\frac{\varepsilon}{2}$ for every $i \in [d]$ or $\frac{f(S)}{\alpha \mathtt{OPT}'_f} > 1-\frac{\varepsilon}{2}$. In the general case of $\alpha, \beta > 0$, if $F'_{\alpha,\beta}(S) > d+1-\frac{\varepsilon}{2}$, we will have $\frac{f(S)}{\alpha \mathtt{OPT}'_f} > 1-\frac{\varepsilon}{2}$ and $\frac{g_i(S)}{\beta \mathtt{OPT}'_{g_i}} > 1-\frac{\varepsilon}{2}$ for every $i \in [d]$ at the same time. Thus, it holds that

$$f(S) \geq (1-\frac{\varepsilon}{2})\alpha \mathtt{OPT}'_f \geq \delta\alpha(1-\frac{\varepsilon}{2})\mathtt{OPT}_f \geq (\delta\alpha-\frac{\varepsilon}{2})\mathtt{OPT}_f$$

and

$$g_i(S) \geq (1-\frac{\varepsilon}{2})\beta \mathtt{OPT}'_{g_i} \geq \delta\beta(1-\frac{\varepsilon}{2})\mathtt{OPT}_{g_i} \geq (\delta\beta-\frac{\varepsilon}{2})\mathtt{OPT}_{g_i}, \forall i \in [d].$$

Therefore, S is a $(\delta\alpha-\frac{\varepsilon}{2}, \delta\beta-\frac{\varepsilon}{2})$-approximate solution to SMFR.

For the proof of the second statement, if $F'_{\alpha,\beta}(S) < d+1$, then we will have $f(S) < \alpha \mathtt{OPT}'_f \leq \alpha \mathtt{OPT}_f$ or there is some $i \in [d]$ with $g_i(S) < \beta \mathtt{OPT}'_{g_i} \leq \beta \mathtt{OPT}_{g_i}$. Therefore, if $F'_{\alpha,\beta}(S) < d+1$, S will not be an (α,β)-approximate solution to SMFR. Consequently, if there is no set $S \in \mathcal{I}$ with $F'_{\alpha,\beta}(S) = d+1$, then there is no (α,β)-approximate solution to SMFR. ☐

A.2 Proof of Theorem 1

Theorem 1. *For SMFR with a knapsack constraint $\mathcal{I}_k$, SMFR-Saturate runs in $O(dt(\mathcal{A}) + \frac{n^2}{\varepsilon}\log\frac{1}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the δ-approximation algorithm for SMK, and provides a set $\mathcal{S}$ of solutions with the following properties: (1) $|\mathcal{S}| = O(\frac{1}{\varepsilon})$, (2) $c(S) = O(k \log \frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$, (3) for each (α^*, β^*)-approximate Pareto optimal solution S^* to SMFR, there must exist its corresponding solution $S \in \mathcal{S}$ such that $f(S) \geq (\delta\alpha^* - \varepsilon)\mathtt{OPT}_f$ and $g_i(S) \geq (\delta\beta^* - \varepsilon)\mathtt{OPT}_{g_i}, \forall i \in [d]$.*

Proof. Let us first analyze the time complexity of SMFR-Saturate for a knapsack constraint $\mathcal{I}_k$. First, it runs the SMK algorithm $d+1$ times to compute $\mathtt{OPT}'_f$ and $\mathtt{OPT}'_{g_i}$ for every $i \in [d]$. Then, it iterates over $\lceil\frac{2}{\varepsilon}\rceil$ values of β in the `for` loop. For each value of β, it attempts to use $O(\log\frac{1}{\varepsilon})$ different values of α in the bisection search. Finally, the subroutine `CostEffectiveGreedy` takes $O(n^2)$ time for SC on each $F'_{\alpha,\beta}$. In summary, the time complexity of SMFR-Saturate for a knapsack constraint $\mathcal{I}_k$ is $O(dt(\mathcal{A}) + \frac{n^2}{\varepsilon}\log\frac{1}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the SMK algorithm.

For the solution $\mathcal{S}$ of SMFR-Saturate, it is easy to see that $|\mathcal{S}| \leq \lceil\frac{2}{\varepsilon}\rceil$ and thus $|\mathcal{S}| = O(\frac{1}{\varepsilon})$ because SMFR-Saturate adds at most one set to $\mathcal{S}$ for each value of β. Then, due to the condition in the `while` loop of the subroutine `CostEffectiveGreedy`, it must hold that $c(S) \leq k(1+\ln\frac{2d+2}{\varepsilon})$ and thus $c(S) = O(k\log\frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$. Finally, given an (α^*, β^*)-approximate Pareto optimal solution S^*, there must exist a value of β in the `for` loop such that $0 \leq \beta^* - \beta \leq \frac{\varepsilon}{2}$. Let $S_{\alpha_{min},\beta}$ be the solution of SMFR-Saturate w.r.t. such β

and its corresponding α_{min}. Since $F'_{\alpha_{min},\beta}(S_{\alpha_{min},\beta}) \geq d+1-\frac{\varepsilon}{2}$, $S_{\alpha_{min},\beta}$ is a $(\delta\alpha_{min}-\frac{\varepsilon}{2},\delta\beta-\frac{\varepsilon}{2})$-approximate solution according to Lemma 1. Furthermore, we have $F'_{\alpha_{max},\beta}(S_{gr}) < d+1-\frac{\varepsilon}{2}$, where S_{gr} is the solution w.r.t. $F'_{\alpha_{max},\beta}$ with a relaxed knapsack constraint for a budget $k(1+\ln\frac{2d+2}{\varepsilon})$ returned by the subroutine `CostEffectiveGreedy` in Algorithm 1, and $\alpha_{max}-\alpha_{min} < \frac{\varepsilon}{2}$. Suppose that S'_{gr} is the first intermediate subset of S_{gr} with $c(S'_{gr}) \geq k\ln\frac{2d+2}{\varepsilon}$ constructed using the cost-effective greedy procedure. Let $S^*_k = \arg\max_{S\in\mathcal{I}_k} F'_{\alpha_{max},\beta}(S)$ and $\mathtt{OPT}_{F'_{\alpha_{max},\beta}} = F'_{\alpha_{max},\beta}(S^*_k)$. According to the monotonicity and submodularity of $F'_{\alpha_{max},\beta}$, we have

$$F'_{\alpha_{max},\beta}(S^*_k) \leq F'_{\alpha_{max},\beta}(S^{(i)}_{gr}) + \sum_{v\in S^*_k\setminus S^{(i)}_{gr}} \Delta(v|S^{(i)}_{gr}) = F'_{\alpha_{max},\beta}(S^{(i)}_{gr}) + \sum_{v\in S^*_k\setminus S^{(i)}_{gr}} \frac{c(v)\cdot\Delta(v|S^{(i)}_{gr})}{c(v)},$$

for any $S^{(i)}_{gr} \subset S'_{gr}$ after i iterations and $\Delta(v|S^{(i)}_{gr}) = F'_{\alpha_{max},\beta}(S^{(i)}_{gr}\cup\{v\}) - F'_{\alpha_{max},\beta}(S^{(i)}_{gr})$. Let u^*_i be the i-th item added to S'_{gr} for any $i=1,\dots,|S'_{gr}|$. Based on the cost-effective greedy selection in Algorithm 1,

$$\frac{\Delta(u^*_{i+1}|S^{(i)}_{gr})}{c(u^*_{i+1})} \geq \frac{\Delta(v|S^{(i)}_{gr})}{c(v)}$$

for any $v\in S^*_k\setminus S^{(i)}_{gr}$ and $i\in[0,\dots,|S'_{gr}-1|]$ because $c(v)\leq k$ for any $v\in S^*_k$ and thus no item from S^*_k is excluded from consideration due to budget violation when u^*_{i+1} is added to $S^{(i)}_{gr}$. Therefore, we further obtain

$$F'_{\alpha_{max},\beta}(S^*_k) \leq F'_{\alpha_{max},\beta}(S^{(i)}_{gr}) + \frac{\Delta(u^*_{i+1}|S^{(i)}_{gr})}{c(u^*_{i+1})}\sum_{v\in S^*_k\setminus S^{(i)}_{gr}} c(v) \leq F'_{\alpha_{max},\beta}(S^{(i)}_{gr}) + \frac{\Delta(u^*_{i+1}|S^{(i)}_{gr})}{c(u^*_{i+1})}\cdot k,$$

After rearranging the inequality above, we have

$$F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i+1)}_{gr}) \leq \big(1-\frac{c(u^*_{i+1})}{k}\big)\big(F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i)}_{gr})\big).$$

Moreover, since $1-x\leq e^{-x}$ for any $x>0$, it holds that $1-\frac{c(u^*_{i+1})}{k} \leq \exp(-\frac{c(u^*_{i+1})}{k})$. Therefore,

$$F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i+1)}_{gr}) \leq \exp(-\frac{c(u^*_{i+1})}{k})\cdot\big(F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i)}_{gr})\big). \tag{4}$$

By applying Eq. 4 recursively to $i=0,\dots|S'_{gr}|-1$, we have

$$\begin{aligned}
F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S'_{gr}) &\leq \exp(-\frac{c(u^*_{i+1})}{k})\cdot\big(F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i)}_{gr})\big)\\
&\leq \exp(-\frac{c(u^*_{i+1})}{k})\exp(-\frac{c(u^*_{i})}{k})\big(F'_{\alpha_{max},\beta}(S^*_k) - F'_{\alpha_{max},\beta}(S^{(i-1)}_{gr})\big)\\
&\leq \dots\dots \leq \exp(-\frac{\sum_{i=0}^{|S'_{gr}|-1} c(u^*_{i+1})}{k})F'_{\alpha_{max},\beta}(S^*_k)\\
&= \exp(-\frac{c(S'_{gr})}{k})F'_{\alpha_{max},\beta}(S^*_k) = \exp(-\frac{c(S'_{gr})}{k})\mathtt{OPT}_{F'_{\alpha_{max},\beta}}.
\end{aligned}$$

Since $c(S'_{gr}) \geq k\ln\frac{2d+2}{\varepsilon}$, it holds that

$$F'_{\alpha_{max},\beta}(S'_{gr}) \geq (1-\exp\big(-\frac{c(S'_{gr})}{k}\big))\mathtt{OPT}_{F'_{\alpha_{max},\beta}} \geq (1-\frac{\varepsilon}{2d+2})\mathtt{OPT}_{F'_{\alpha_{max},\beta}}.$$

In addition, $F'_{\alpha_{max},\beta}(S_{gr}) \geq F'_{\alpha_{max},\beta}(S'_{gr})$ since $S'_{gr}\subseteq S_{gr}$. Therefore, we have $\mathtt{OPT}_{F'_{\alpha_{max},\beta}} < d+1$ and, according to Lemma 1, there does not exist any (α_{max},β)-approximate solution of cost at most k. Since S^* is an (α^*,β^*)-approximate Pareto optimal solution and $\beta\leq\beta^*$, S^* must be an (α^*,β)-approximate solution of cost at most k. As such, we obtain $\alpha_{max}>\alpha^*$ and $\alpha_{min}>\alpha^*-\frac{\varepsilon}{2}$. Because we have shown that $S_{\alpha_{min},\beta}$ is a $(\delta\alpha_{min}-\frac{\varepsilon}{2},\delta\beta-\frac{\varepsilon}{2})$-approximate solution, $S_{\alpha_{min},\beta}$ is guaranteed to be a $(\delta\alpha^*-\varepsilon,\delta\beta^*-\varepsilon)$-approximate solution. If $S_{\alpha_{min},\beta}$ is included in $\mathcal{S}$, we will conclude the proof directly; otherwise, the solution in $\mathcal{S}$ dominating $S_{\alpha_{min},\beta}$ can confirm our conclusion. □

A.3 Proof of Theorem 2

Theorem 2. *For SMFR with a matroid constraint $\mathcal{I}(\mathcal{M})$, SMFR-Saturate runs in $O(dt(\mathcal{A}) + \frac{nr}{\varepsilon}\log^2\frac{d}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the δ-approximation algorithm for SMM, and provides a set $\mathcal{S}$ of solutions with the following properties: (1) $|\mathcal{S}| = O(\frac{1}{\varepsilon})$, (2) $|S| = O(r\log\frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$, (3) for each (α^*, β^*)-approximate Pareto optimal solution S^* to SMFR, there must exist its corresponding solution $S \in \mathcal{S}$ such that $f(S) \geq (\delta\alpha^* - \varepsilon)\mathtt{OPT}_f$ and $g_i(S) \geq (\delta\beta^* - \varepsilon)\mathtt{OPT}_{g_i}, \forall i \in [d]$.*

Proof. Let us analyze the time complexity of SMFR-Saturate for a matroid constraint $\mathcal{I}(\mathcal{M})$. First, it runs the SMM algorithm $d+1$ times to compute $\mathtt{OPT}'_f$ and $\mathtt{OPT}'_{g_i}$ for every $i \in [d]$. Then, it iterates over $\lceil\frac{2}{\varepsilon}\rceil$ values of β in the `for` loop. For each value of β, it attempts to use $O(\log\frac{1}{\varepsilon})$ different values of α in the bisection search. Finally, the subroutine `IterativeGreedy` takes $O(nr)$ time per round and runs in $O(\log\frac{d}{\varepsilon})$ rounds. In summary, the time complexity of SMFR-Saturate for a matroid constraint $\mathcal{I}(\mathcal{M})$ is $O(dt(\mathcal{A}) + \frac{nr}{\varepsilon}\log\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$ time, where $t(\mathcal{A})$ is the time complexity of the SMM algorithm, and can be simplified as $O(dt(\mathcal{A}) + \frac{nr}{\varepsilon}\log^2\frac{d}{\varepsilon})$.

For the solution $\mathcal{S}$ of SMFR-Saturate, it is easy to see that $|\mathcal{S}| \leq \lceil\frac{2}{\varepsilon}\rceil$ and thus $|\mathcal{S}| = O(\frac{1}{\varepsilon})$ because SMFR-Saturate adds at most one set to $\mathcal{S}$ for each value of β. Then, because the subroutine `IterativeGreedy` runs in at most $1 + \lceil\log_2\frac{d+1}{\varepsilon}\rceil$ rounds and the size of each S_l is bounded by the rank r of the matroid $\mathcal{M}$, it must hold that $|S| \leq r \cdot (1 + \lceil\log_2\frac{d+1}{\varepsilon}\rceil)$ and thus $|S| = O(r\log\frac{d}{\varepsilon})$ for each $S \in \mathcal{S}$. Finally, given an (α^*, β^*)-approximate Pareto optimal solution S^*, there must exist a value of β in the `for` loop such that $0 \leq \beta^* - \beta \leq \frac{\varepsilon}{2}$. Let $S_{\alpha_{min},\beta}$ be the solution of SMFR-Saturate w.r.t. such β and its corresponding α_{min}. Since $F'_{\alpha_{min},\beta}(S_{\alpha_{min},\beta}) \geq d + 1 - \frac{\varepsilon}{2}$, $S_{\alpha_{min},\beta}$ is a $(\delta\alpha_{min} - \frac{\varepsilon}{2}, \delta\beta - \frac{\varepsilon}{2})$-approximate solution according to Lemma 1. Furthermore, we have $F'_{\alpha_{max},\beta}(S_{gr}) < d + 1 - \frac{\varepsilon}{2}$, where S_{gr} is the solution w.r.t. $F'_{\alpha_{max},\beta}$ returned by the subroutine `IterativeGreedy` in Algorithm 1, and $\alpha_{max} - \alpha_{min} < \frac{\varepsilon}{2}$. Since `IterativeGreedy` runs a $\frac{1}{2}$-approximation greedy algorithm for submodular maximization with matroid constraints in each round, we have

$$F'_{\alpha_{max},\beta}(S_1) - F'_{\alpha_{max},\beta}(\emptyset) \geq \big(1 - \frac{1}{2}\big) \cdot \max_{S' \in \mathcal{I}(\mathcal{M})}(F'_{\alpha_{max},\beta}(S') - F'_{\alpha_{max},\beta}(\emptyset)).$$

Since $\widetilde{f}(S) = f(S \cup A) - f(S)$ is nonnegative, monotone, and submodular if $f(\cdot)$ is nonnegative, monotone, and submodular for any $A \subseteq V$, we can extend the above result for each round $l > 1$ as follows:

$$\begin{aligned}F'_{\alpha_{max},\beta}(\cup_{j=1}^{l}S_j) - F'_{\alpha_{max},\beta}(\cup_{j=1}^{l-1}S_j) &\geq \big(1 - \frac{1}{2}\big) \cdot \max_{S'_l \in \mathcal{I}(\mathcal{M})}(F'_{\alpha_{max},\beta}(S'_l \cup (\cup_{j=1}^{l-1}S_j)) - F'_{\alpha_{max},\beta}(\cup_{j=1}^{l-1}S_j)) \\ &\geq \big(1 - \frac{1}{2}\big) \cdot \max_{S' \in \mathcal{I}(\mathcal{M})}(F'_{\alpha_{max},\beta}(S') - F'_{\alpha_{max},\beta}(\cup_{j=1}^{l-1}S_j)).\end{aligned}$$

By induction, we obtain the following:

$$F'_{\alpha_{max},\beta}(\cup_{j=1}^{l}S_j) \geq \big(1 - \frac{1}{2^l}\big) \cdot \max_{S' \in \mathcal{I}(\mathcal{M})} F'_{\alpha_{max},\beta}(S') = \big(1 - \frac{1}{2^l}\big)\mathtt{OPT}_{F'_{\alpha_{max},\beta}}.$$

Since $S_{gr} = \cup_{j=1}^{1+\lceil\log_2\frac{d+1}{\varepsilon}\rceil} S_j$, we have

$$F'_{\alpha_{max},\beta}(S_{gr}) \geq \big(1 - \frac{1}{2^l}\big) \cdot \mathtt{OPT}_{F'_{\alpha_{max},\beta}} \geq \big(1 - \frac{\varepsilon}{2d+2}\big) \cdot \mathtt{OPT}_{F'_{\alpha_{max},\beta}}.$$

Therefore, we have $\mathtt{OPT}_{F'_{\alpha_{max},\beta}} < d + 1$ and, according to Lemma 1, there does not exist any (α_{max}, β)-approximate solution under matroid constraint $\mathcal{I}(\mathcal{M})$. Since S^* is an (α^*, β^*)-approximate Pareto optimal solution and $\beta \leq \beta^*$, S^* must be an (α^*, β)-approximate solution under matroid constraint $\mathcal{I}(\mathcal{M})$. As such, we obtain $\alpha_{max} > \alpha^*$ and $\alpha_{min} > \alpha^* - \frac{\varepsilon}{2}$. Because we have shown that $S_{\alpha_{min},\beta}$ is a $(\delta\alpha_{min} - \frac{\varepsilon}{2}, \delta\beta - \frac{\varepsilon}{2})$-approximate solution, $S_{\alpha_{min},\beta}$ is guaranteed to be a $(\delta\alpha^* - \varepsilon, \delta\beta^* - \varepsilon)$-approximate solution. If $S_{\alpha_{min},\beta}$ is included in $\mathcal{S}$, we will conclude the proof directly; otherwise, the solution in $\mathcal{S}$ dominating $S_{\alpha_{min},\beta}$ can confirm our conclusion. □

B ILP formulations

In this section, we present the integer linear programming (ILP) formulations for the *Maximum Coverage* and *Recommendation* problems, specifically tailored to the SMFR problem, as defined in Section 5.1 and Section 5.2, respectively. Any ILP solver can be employed to identify optimal solutions for small SMFR instances on *Maximum Coverage* and *Recommendation.* For our experimental results in Section 5 and Appendix C, we refer to this approach as the OPT algorithm. Note that these formulations are specifically designed for these settings and cannot be applied directly to general SMFR problems.

Problems 5 and 6 are specialized versions of the standard ILP formulation of SMFR on *Maximum Coverage*[6] in Section 5.1, with knapsack and partition matroid constraints, respectively.

$$
\begin{array}{lll}
\max & \sum_{j\in[m]} y_j & \\
\text{subject to} & \sum_{l\in[n]} c_l x_l \le k & \\
& \sum_{e_j\in S_l} x_l \ge y_j, & \forall j\in[m] \\
& \sum_{e_j\in C_i} y_j \ge \beta\,\mathtt{OPT}_{g_i}, & \forall i\in[d] \\
& y_j\in\{0,1\}, & \forall j\in[m] \\
& x_l\in\{0,1\}, & \forall l\in[n]
\end{array}
\tag{5}
$$

$$
\begin{array}{lll}
\max & \sum_{j\in[m]} y_j & \\
\text{subject to} & \sum_{S_l\in V_t} x_l \le k, & \forall t\in[p] \\
& \sum_{e_j\in S_l} x_l \ge y_j, & \forall j\in[m] \\
& \sum_{e_j\in C_i} y_j \ge \beta\,\mathtt{OPT}_{g_i}, & \forall i\in[d] \\
& y_j\in\{0,1\}, & \forall j\in[m] \\
& x_l\in\{0,1\}, & \forall l\in[n]
\end{array}
\tag{6}
$$

These ILPs maximize the *coverage* (i.e., the utility function f in SMFR) on a universe $U=\{e_1,\ldots,e_m\}$ of m elements and a collection $V=\{S_1,\ldots,S_n\}$ of n sets ($S_l\subseteq V, \forall l\in[n]$), subject to additional coverage constraints on each subset $C_1,\ldots,C_d$ of U (w.r.t. each representativeness function $g_1,\ldots,g_d$ in SMFR). In both formulations, x_l indicates whether $S_l\in V$ is included in the solution S, and y_j indicates whether $e_j\in U$ is covered by S. Problem 5 is specific to the knapsack constraint defined on a budget $k\in\mathbb{Z}^+$ and a cost function $c(\cdot)$. Problem 6 is specific to the partition matroid constraint, where V is divided into p disjoint partitions $V_1,\ldots,V_p$ and at most k sets can be selected from each partition. Solving optimally Problems 5 and 6 with $\beta=0$ and $U=C_i$ yields the value of $\mathtt{OPT}_{g_i}$ for each representativeness function g_i corresponding to the knapsack and the partition matroid constraints, respectively.

Problems 7 and 8 are specialized versions of the ILP formulation for capacitated facility location[7], with a benefit matrix $B=\{b_{jl}=\langle\mathbf{v}_i,\mathbf{v}_j\rangle : j\in[m], l\in[n]\}\in\mathbb{R}^{m\times n}$ ($m=|T|$ and $n=|V|$), specifically designed for SMFR on the *Recommendation* setting in Section 5.2, with knapsack and partition matroid constraints, respectively.

$$
\begin{array}{lll}
\max & \sum_{j\in[m]}\sum_{l\in[n]} b_{jl} y_{jl} & \\
\text{subject to} & \sum_{l\in[n]} c_l x_l \le k & \\
& \sum_{l\in[n]} y_{jl} \le 1, & \forall j\in[m] \\
& y_{jl} \le x_l, & \forall j\in[m], l\in[n] \\
& \sum_{e_l\in C_i} x_l \ge \beta\,\mathtt{OPT}_{g_i}, & \forall i\in[d] \\
& y_{jl}\in\{0,1\}, & \forall j\in[m], l\in[n] \\
& x_l\in\{0,1\}, & \forall l\in[n]
\end{array}
\tag{7}
$$

$$
\begin{array}{lll}
\max & \sum_{j\in[m]}\sum_{l\in[n]} b_{jl} y_{jl} & \\
\text{subject to} & \sum_{e_l\in V_t} x_l \le k, & \forall t\in[p] \\
& \sum_{l\in[n]} y_{jl} \le 1, & \forall j\in[m] \\
& y_{jl} \le x_l, & \forall j\in[m], l\in[n] \\
& \sum_{e_l\in C_i} x_l \ge \beta\,\mathtt{OPT}_{g_i}, & \forall i\in[d] \\
& y_{jl}\in\{0,1\}, & \forall j\in[m], l\in[n] \\
& x_l\in\{0,1\}, & \forall l\in[n]
\end{array}
\tag{8}
$$

---

[6] `https://en.wikipedia.org/wiki/Maximum_coverage_problem`

[7] `https://en.wikipedia.org/wiki/Optimal_facility_location`

Given a set $V = \{e_1, \dots, e_n\}$ of n items, both ILPs maximize the total *benefit* (i.e., the utility function f in SMFR) provided by a set $S \subseteq V$ for a subset $T \subseteq V$ of m items, subject to representativeness constraints on each $C_1, \dots, C_d$ subset of V (i.e., the representativeness functions $g_1, \dots, g_d$ in SMFR). In both formulations, x_l indicates whether $e_l \in V$ is included in the solution S, and y_{jl} indicates whether $e_j \in T$ takes the benefit from item $e_l \in V$. Problem 7 is specific to the knapsack constraint defined on a budget $k \in \mathbb{Z}^+$ and a cost function $c(\cdot)$. Problem 8 is specific to the partition matroid constraint, where V is divided into p disjoint partitions $V_1, \dots, V_p$. For the knapsack constraint, the value of $\mathtt{OPT}_{g_i}$ for each representativeness function g_i can be easily computed by sorting the items in C_i ascendingly according to their costs and finding the maximum number of items whose cumulative cost does not exceed k. For the partition matroid constraint, the value of $\mathtt{OPT}_{g_i}$ for each representativeness function g_i is trivially the maximum between k and $|C_i|$.

C Additional Experiments

In this section, we complement the experimental analysis described in Sections 5.1 and 5.2.

C.1 Additional Experiments on Maximum Coverage

In this section, we use the same datasets and settings as in Section 5.1 for the *Maximum Coverage* problem. For the *Facebook* dataset, we alternatively define the knapsack constraint in the same way as for the *DBLP* dataset. For the *DBLP* dataset, we alternatively define a partition matroid constraint based on the geographic area of the researchers, with five groups: *Asia*, *Europe*, *North America*, *Oceania*, and *South America*. We also set the same upper bound $k \in \mathbb{Z}^+$ for each geographic group, resulting in a partition matroid of rank $r = 5k$. Figures 5 and 6 present the performance of each algorithm for different instances of SMFR on *Maximum Coverage* with knapsack and matroid constraints on the *Facebook* and *DBLP* datasets, respectively. Generally, we observe trends similar to those already presented in Section 5.1, which further confirm the effectiveness of SMFR-Saturate.

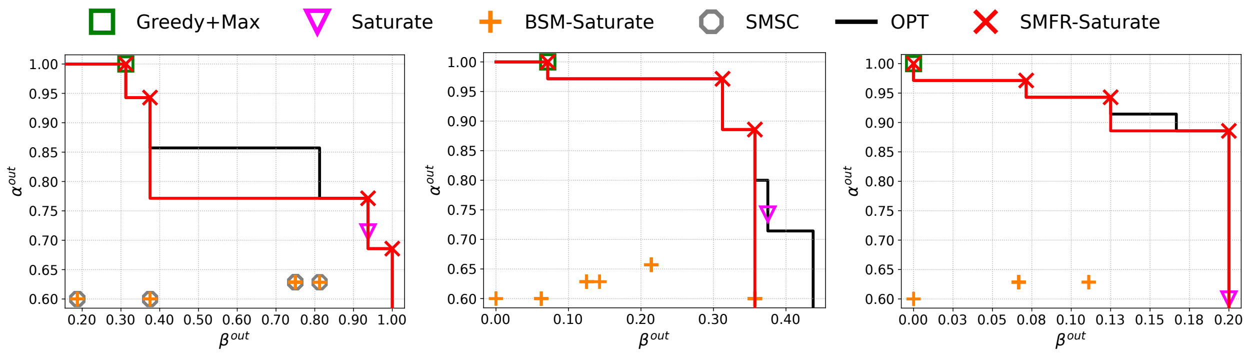
C.2 Additional Experiments on Recommendation

In this section, we use the same datasets and settings as in Section 5.2 for the *Recommendation* problem. For the *MovieLens* dataset, we alternatively define a knapsack constraint by assigning to each item (movie) a random integer cost in the range $[1, 10]$. For the *X-Wines* dataset, we alternatively define a partition matroid constraint based on the continent of origin for wine production: *Africa*, *Asia*, *Europe*, *North America*, *South America*, and *Oceania*. We also set the same upper bound $k \in \mathbb{Z}^+$ for each geographic group, resulting in a partition matroid of rank $r = 6k$. Figures 7 and 8 present the performance of each algorithm for different instances of SMFR on *Recommendation* with knapsack and matroid constraints on the *MovieLens* and *X-Wines* datasets, respectively. Generally, we observe trends similar to those already presented in Section 5.2, which further confirm the effectiveness of SMFR-Saturate.

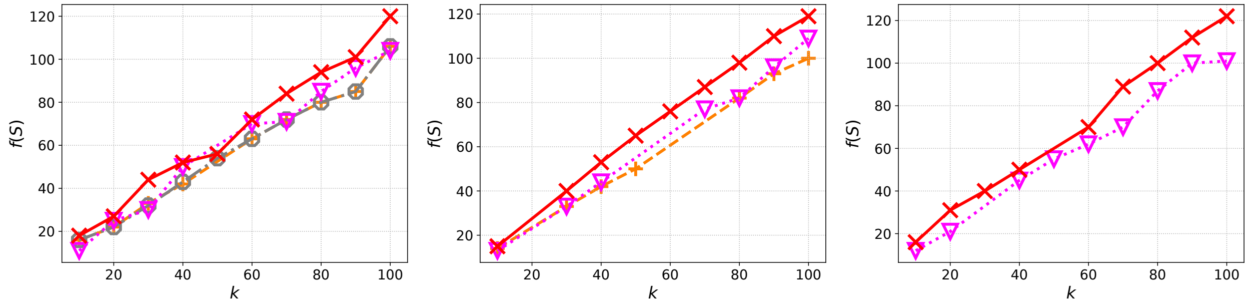
C.3 Time Efficiency

Figure 9 reports the running time (in seconds) of SMFR-Saturate, Saturate, BSM-Saturate, and SMSC for SMFR on both *Maximum Coverage* and *Recommendation* instances. We use the same settings as in Sections 5.1 and 5.2. In each plot, the x-axis represents the value of k in the knapsack or matroid constraint and the y-axis represents the running time (in seconds) used by each algorithm to solve an SMFR instance. We present the results for $d = 1$ and 4 in Figure 9.

All algorithms take less than a minute to complete on each tested instance. SMFR-Saturate is faster than SMSC in all cases. For the knapsack constraints, SMFR-Saturate generally runs faster than or close to BSM-Saturate. However, for the matroid constraints, SMFR-Saturate is slower than BSM-Saturate. Saturate is the fastest method in most configurations. This is because Saturate does not allow for any trade-off between utility (f) and representativeness (g) by design and thus is run only once for each instance. However, all other algorithms should be run multiple times with different values of β or τ.

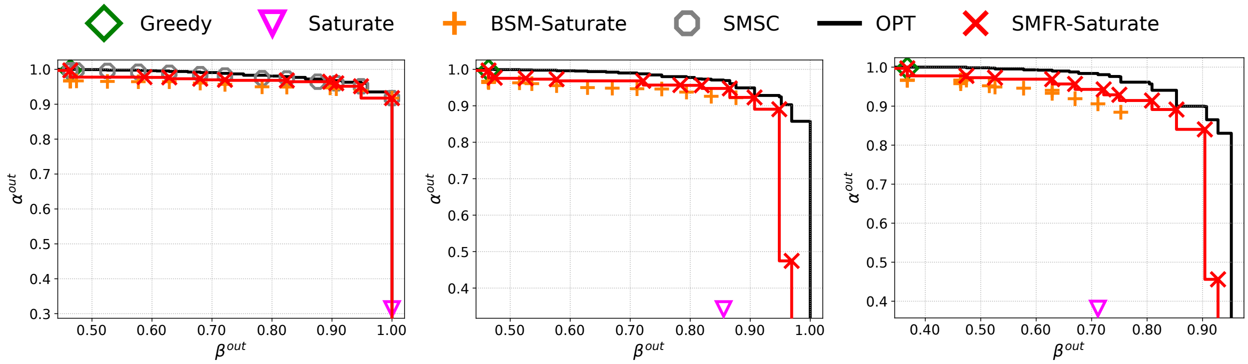


(a) *Facebook* $(d = 1, k = 20)$ (b) *Facebook* $(d = 2, k = 20)$ (c) *Facebook* $(d = 4, k = 20)$

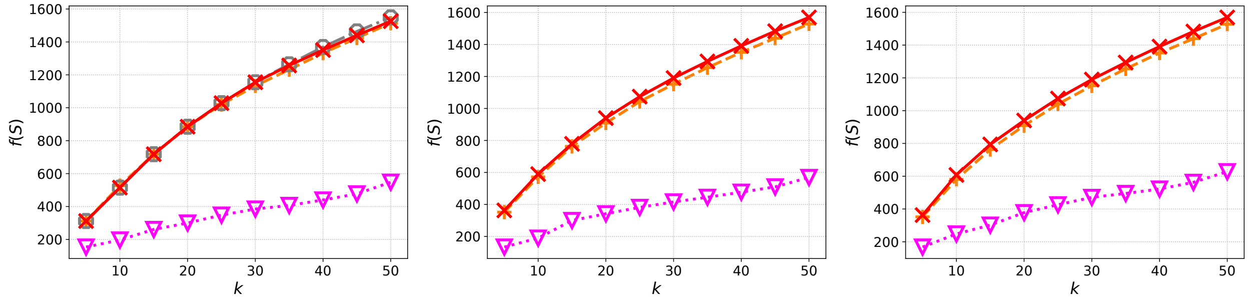


(d) *Facebook* $(d = 1, \beta \geq 0.8)$ (e) *Facebook* $(d = 2, \beta \geq 0.4)$ (f) *Facebook* $(d = 4, \beta \geq 0.2)$

Figure 5: Results for *Maximum Coverage* on the *Facebook* dataset, with knapsack constraints.

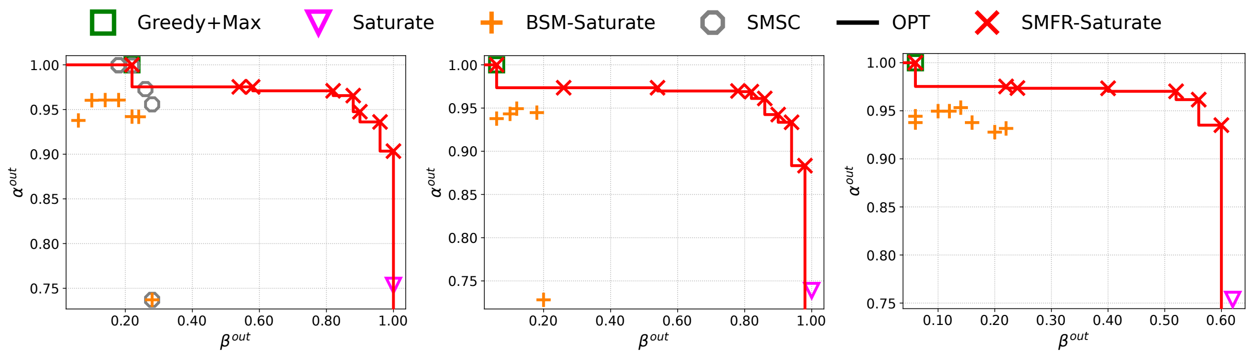


(a) *DBLP* $(d = 1, k = 35)$ (b) *DBLP* $(d = 2, k = 35)$ (c) *DBLP* $(d = 4, k = 35)$

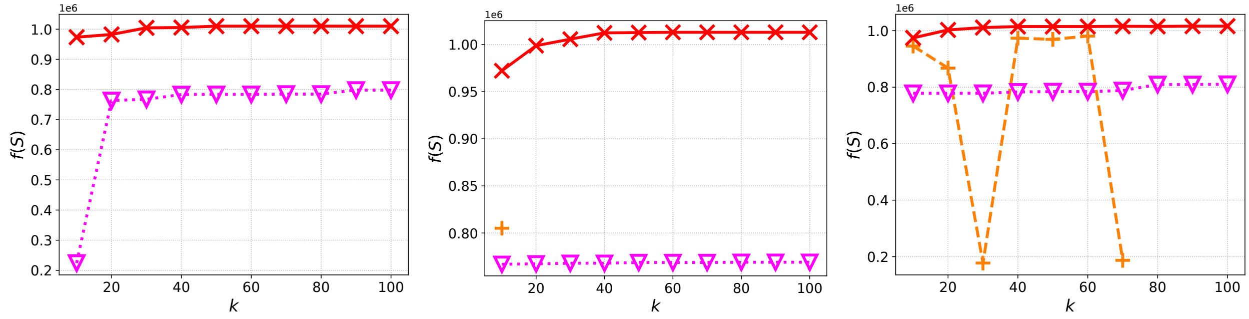


(d) *DBLP* $(d = 1, \beta \geq 0.8)$ (e) *DBLP* $(d = 2, \beta \geq 0.4)$ (f) *DBLP* $(d = 4, \beta \geq 0.2)$

Figure 6: Results for *Maximum Coverage* on the *DBLP* dataset, with matroid constraints.

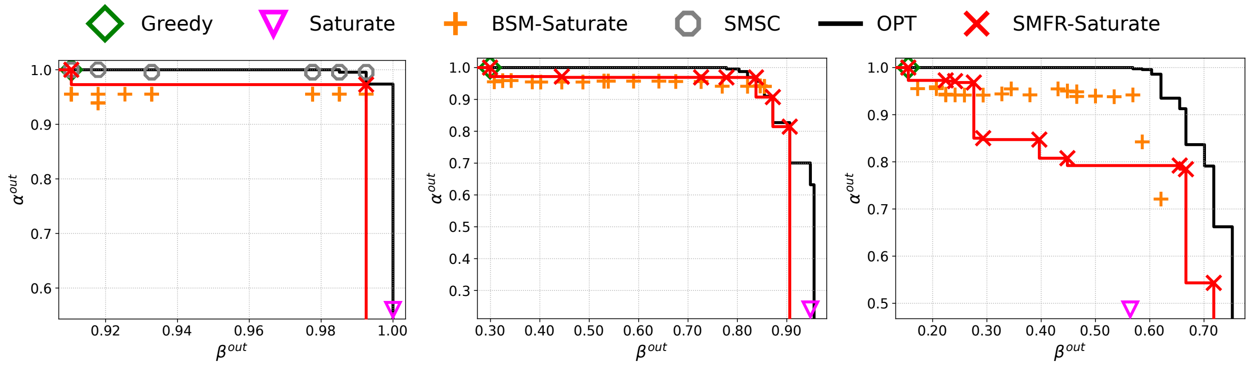


(a) *MovieLens* $(d = 1, k = 50)$ (b) *MovieLens* $(d = 2, k = 50)$ (c) *MovieLens* $(d = 4, k = 50)$

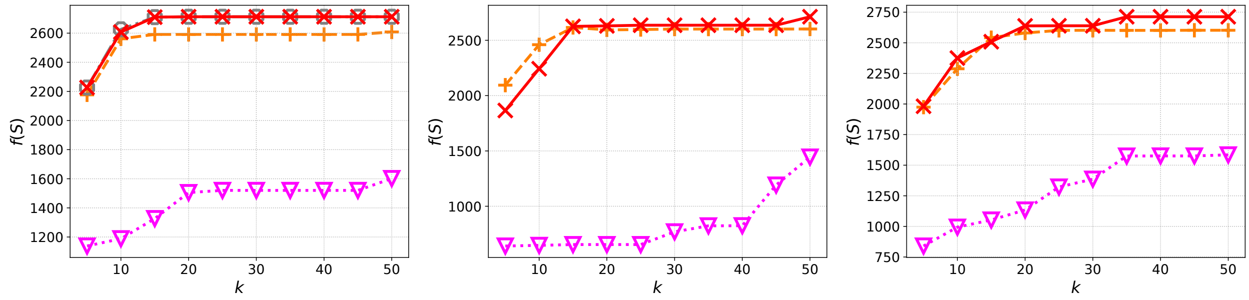


(d) *MovieLens* $(d = 1, \beta \geq 0.8)$ (e) *MovieLens* $(d = 2, \beta \geq 0.4)$ (f) *MovieLens* $(d = 4, \beta \geq 0.2)$

Figure 7: Results for *Recommendation* on the *MovieLens* dataset, with knapsack constraints.

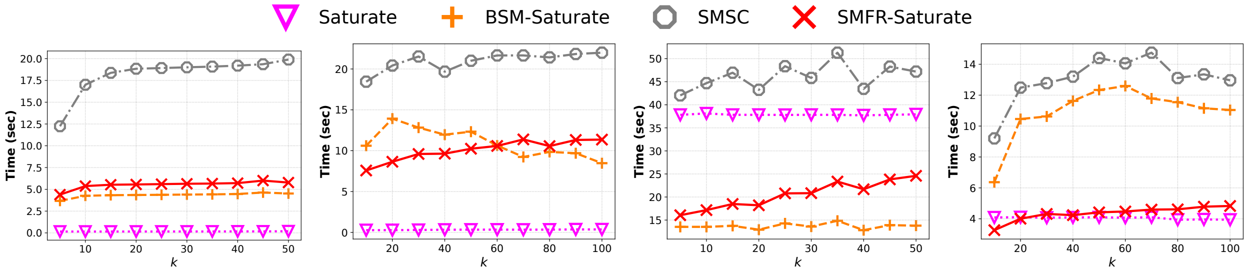


(a) *X-Wines* $(d = 1, k = 25)$ (b) *X-Wines* $(d = 2, k = 25)$ (c) *X-Wines* $(d = 4, k = 25)$

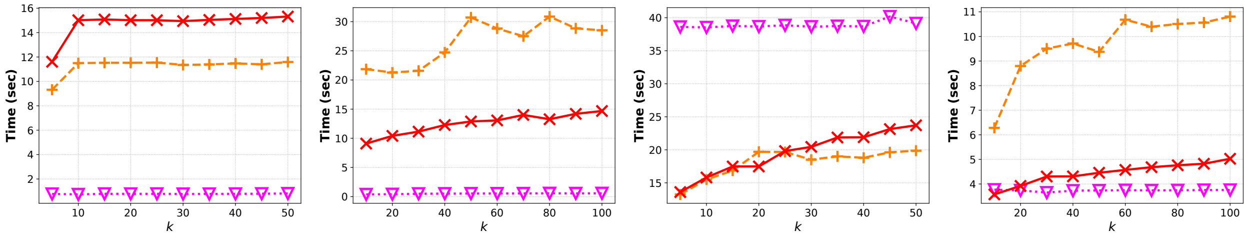


(d) *X-Wines* $(d = 1, \beta \geq 0.8)$ (e) *X-Wines* $(d = 2, \beta \geq 0.4)$ (f) *X-Wines* $(d = 4, \beta \geq 0.2)$

Figure 8: Results for *Recommendation* on the *X-Wines* dataset, with matroid constraints.



(a) *Facebook*, $d = 1$, Mat. (b) *DBLP*, $d = 1$, Kna. (c) *MovieLens*, $d = 1$, Mat. (d) *X-Wines*, $d = 1$, Kna.



(e) *Facebook*, $d = 4$, Mat. (f) *DBLP*, $d = 4$, Kna. (g) *MovieLens*, d=4, Mat. (h) *X-Wines*, $d = 4$, Kna.

Figure 9: Running times (in seconds) of SMFR-Saturate, Saturate, BSM-Saturate, and SMSC for SMFR when $d = 1, 4$). Here, the *Facebook* and *DBLP* datasets are used for *Maximum Coverage* (MC); the *X-Wines* and *MovieLens* datasets are used for *Recommendation* (RE). In addition, the matroid constraints (Mat.) are imposed on the *Facebook* and *MovieLens* datasets; the knapsack constraints (Kna.) are imposed on the *DBLP* and *X-Wines* datasets.