# Maximizing Submodular Set Functions Subject to Multiple Linear Constraints

Ariel Kulik* Hadas Shachnai† Tami Tamir ‡

**Abstract**

The concept of submodularity plays a vital role in combinatorial optimization. In particular, many important optimization problems can be cast as submodular maximization problems, including maximum coverage, maximum facility location and max cut in directed/undirected graphs.

In this paper we present the first known approximation algorithms for the problem of maximizing a non-decreasing submodular set function subject to multiple linear constraints. Given a $d$-dimensional budget vector $\bar{L}$, for some $d \geq 1$, and an oracle for a non-decreasing submodular set function $f$ over a universe $U$, where each element $e \in U$ is associated with a $d$-dimensional cost vector, we seek a subset of elements $S \subseteq U$ whose total cost is at most $\bar{L}$, such that $f(S)$ is maximized.

We develop a framework for maximizing submodular functions subject to $d$ linear constraints that yields a $(1 - \varepsilon)(1 - e^{-1})$-approximation to the optimum for any $\varepsilon > 0$, where $d > 1$ is some constant. Our study is motivated by a variant of the classical maximum coverage problem that we call *maximum coverage with multiple packing constraints*. We use our framework to obtain the same approximation ratio for this problem. To the best of our knowledge, this is the first time the theoretical bound of $1 - e^{-1}$ is (almost) matched for both of these problems.

## 1 Introduction

A function $f$, defined over a collection of subsets of a universe $U$, is called *submodular* if, for any $S, T \subseteq U$,

$$f(S) + f(T) \geq f(S \cup T) + f(S \cap T).$$

Alternatively, $f$ is submodular if it satisfies the property of *decreasing marginal value*, namely, for any $A \subseteq B \subseteq U$ and $e \in U \setminus B$,

$$f(B \cup \{e\}) - f(B) \leq f(A \cup \{e\}) - f(A).$$

The function $f$ is *non-decreasing* if, for any subsets $T$ and $S$ such that $T \subseteq S$, $f(T) \leq f(S)$. The concept of submodularity plays a vital role in combinatorial theorems and algorithms, and its importance in discrete optimization has been well studied (see, e.g., [7] and the references therein, and the surveys in [5, 16]). Submodularity can be viewed as a discrete analog of convexity. Many practically important optimization problems, including maximum coverage, maximum facility location, and max cut in directed/undirected graphs, can be cast as submodular optimization problems (see, e.g., [5]).

This paper presents the first known approximation algorithms for the problem of maximizing a non-decreasing submodular set function subject to multiple linear constraints. Given a $d$-dimensional budget vector $\bar{L}$, for some $d \geq 1$, and an oracle for a non-decreasing submodular set function $f$ over a universe $U$, where each element $i \in U$ is associated with a $d$-dimensional cost vector $\bar{c}_i$, we seek a subset of elements $S \subseteq U$ whose total cost is at most $\bar{L}$, such that $f(S)$ is maximized.

There has been extensive work on maximizing submodular monotone functions subject to matroid constraint.[1] For the special case of *uniform matroid*, i.e., the problem $\{\max f(S) : |S| \leq k\}$, for some $k > 1$, Nemhauser et. al showed in [11] that a Greedy algorithm yields a ratio of $1 - e^{-1}$ to the optimum. Later works presented Greedy algorithms that achieve this ratio for other special matroids or for certain submodular monotone functions (see, e.g., [1, 9, 15, 3]). For a general matroid constraint, Calinescu et al. showed in [2] that a scheme based on solving a continuous relaxation of the problem followed by *pipage rounding* (a technique introduced by Ageev and Sviridenko [1]) achieves the ratio of $1 - e^{-1}$ for maximizing submodular monotone functions that can be expressed as a sum of weighted rank functions of matroids. Recently, this result was extended by Vondrák [16] to general monotone submodular functions. The bound of $1 - e^{-1}$ is the best possible for all of the above problems; this follows from a result of Feige [4], which holds already for the maximum coverage problem.

*Computer Science Dept., Technion, Haifa 32000, Israel. E-mail: kulik@cs.technion.ac.il

†Computer Science Dept., Technion, Haifa 32000, Israel. E-mail: hadas@cs.technion.ac.il. Work supported by the Technion V.P.R. Fund.

‡School of Computer Science, The Interdisciplinary Center, Herzliya, Israel. E-mail: tami@idc.ac.il

[1] A (weighted) matroid is a system of 'independent subsets' of a universe, which satisfies certain *hereditary* and *exchange* properties [12].

The techniques introduced in these previous works are powerful and elegant, but do not seem to lead to efficient approximation algorithms for maximizing a submodular function subject to $d$ linear constraints, already for $d = 2$. While the Greedy algorithm is undefined for $d > 1$, a major difficulty in rounding the solution of the continuous problem (as in [2, 16]) is to preserve the approximation ratio while satisfying the constraints. A noteworthy contribution of our framework is in finding a way to get around this difficulty (see Section 1.1).

Our study is motivated by the following variant of the classical maximum coverage problem that we call *maximum coverage with multiple packing constraints* (MCMP). Given is a collection of subsets $\{S_1, \ldots, S_m\}$ over a ground set of elements $A = \{a_1, \ldots, a_n\}$. Each element $a_j$ is associated with a $d$-dimensional size vector $\bar{s}_j = (s_{j,1}, \ldots, s_{j,d})$ and a non-negative value $w_j$. Also, given is a $d$-dimensional bin whose capacity is $\bar{B} = (B_1, \ldots, B_d)$, and a budget $k > 1$. The goal is to select $k$ subsets in $\{S_1, \ldots, S_m\}$ and determine which of the elements in these subsets are *covered*, such that the overall size of covered elements is at most $\bar{B}$, and their total value is maximized. In the special case where $d = 1$, we call the problem *maximum coverage with packing constraint* (MCP).

MCP is known to be APX-hard, even if all elements have the same (unit) size and the same (unit) profit, and each element belongs to at most four subsets [13]. Since MCP includes as a special case the maximum coverage problem, the best approximation ratio one can expect is $1 - e^{-1}$ [4].[2]

### 1.1 Our Results

In Section 2 we develop a framework for maximizing submodular functions subject to $d$ linear constraints, that yields a $(1 - \varepsilon)(1 - e^{-1})$-approximation to the optimum for any $\varepsilon > 0$, where $d > 1$ is some constant. This extends a result of [15] (within factor $1 - \varepsilon$). A key component in our framework is to obtain approximate solution for a continuous relaxation of the problem. This can be done using an algorithm recently presented by Vondrák [16]. For some specific submodular functions, other techniques can be used to obtain fractional solutions with the same properties (see, e.g., [1, 2]).

In Section 3 we show that MCP can be approximated within factor $1 - e^{-1}$, by applying known results for maximizing submodular functions. Here we use the fact that the *fractional* version of MCP defines a non-decreasing submodular set function; this is not true already for $d = 2$. For MCMP we show (in Section 4) that our framework yields an approximation ratio of $(1 - \varepsilon)(1 - e^{-1})$ when $d > 1$ is a constant.

**Technical Contribution:** The heart of our framework is a rounding step that preserves multiple linear constraints. Here we use a non-trivial combination of randomized rounding with two enumeration phases: one on the most profitable elements in some optimal solution, and the other on the 'big' elements (see in Section 2). This enables to show that the rounded solution can be converted to a feasible one with high expected profit.

Due to space constraints, some of the proofs are omitted. The detailed results appear in [10].

## 2 Maximizing Submodular Functions

In this section we describe our framework for maximizing a non-decreasing submodular set function subject to multiple constraints. For short, we call this problem MLC.

### 2.1 Preliminaries

Given a universe $U$, we call a subset of elements $S \subseteq U$ *feasible* if the total cost of elements in $S$ is bounded by $\bar{L}$; we refer to $f(S)$ as the value of $f$.

An essential component in our framework is the distinction between elements by their costs. We say that an element $i \in U$ is *big in dimension* $r$ if $c_{i,r} \geq \varepsilon^4 L_r$; element $i$ is *big* if for some $1 \leq r \leq d$, $i$ is big in dimension $r$. An element is *small in dimension* $r$ if it is not big in dimension $r$, and *small* if it is not big. Note that the number of big elements in a feasible solution is at most $d \cdot \varepsilon^{-4}$.

Our framework applies some preliminary steps, after which it solves a *residual problem*. Given an instance of MLC, we consider two types of residual problems. For a subset $T \subseteq U$, define another instance of MLC in which the objective function is $f_T(S) = f(S \cup T) - f(T)$ (it is easy to verify that $f_T$ is a non-decreasing submodular set function); the cost of each element remains as in the original instance, the budget is $\bar{L} - \bar{c}(T)$ where $\bar{c}(T) = \sum_{i \in T} \bar{c}_i$, and the universe (which is a subset of the original universe) depends on the type of residual problem.

- *Value residual problem-* the universe consists of all elements $i \in U \setminus T$ such that $f_T(\{i\}) \leq \frac{f(T)}{|T|}$.
- *Cost residual problem-* the universe consists of all *small* elements in the original problem.

These two types of problems allow us to convert the original problem to a problem with some desired properties, namely, either all elements are of bounded value, or all elements are of bounded cost in each dimension.

[2] For other known results for the maximum coverage problem, see e.g., [9, 14, 1].

**Extension by Expectation:** Given a non-decreasing submodular function $f : 2^U \rightarrow \mathbb{R}_+$, we define $F : [0,1]^U \rightarrow \mathbb{R}_+$ to be the following continuous extension of $f$. For any $\bar{y} \in [0,1]^U$, let $R \subseteq U$ be a random variable such that $i \in R$ with probability $y_i$. Then define

$$F(\bar{y}) = E[f(R)] = \sum_{R \subseteq U} \left( f(R) \prod_{i \in R} y_i \prod_{i \notin R} (1 - y_i) \right)$$

(For the submodular function $f_T$, the continuous extension is denoted by $F_T$.) This extension of a submodular function has been previously studied (see, e.g, [1, 2, 16]). We consider the following continuous relaxation of MLC. Define the polytope of the instance

$$P = \{\bar{y} \in [0,1]^U | \sum_{i \in U} y_i \bar{c}_i \leq \bar{L}\},$$

and the problem is to find $\bar{y} \in P$ for which $F(\bar{y})$ is maximized. Similar to the discrete case, $\bar{y} \in [0,1]^U$ is feasible if $\bar{y} \in P$.

For some specific submodular functions, linear programming can be used to obtain $\bar{y} \in P$ such that $F(\bar{y}) \geq (1 - e^{-1})\mathcal{O}$, where $\mathcal{O}$ is the optimal solution for MLC (see e.g [1, 2]). Recently, Vondrák [16] gave an algorithm that finds $\bar{y} \in P'$ such that $F(\bar{y}) \geq (1 - e^{-1} - o(1))\mathcal{O}_f$ where $\mathcal{O}_f = \max_{\bar{z} \in P'} F(\bar{z}) \geq \mathcal{O}$, and $P'$ is a matroid polytope.[3] While the algorithm of [16] is presented in the context of matroid polytopes, it can be easily extended to general convex polytope $P$ with $\bar{0} \in P$, as long as the value $\bar{y} = \text{argmax}_{\bar{y} \in P} \sum_{i \in U} y_i w_i$ can be efficiently found for any vector $\bar{w} \in \mathbb{R}_+^U$. In our case, this can be efficiently done using linear programming. The algorithm of [16] can be used in our framework for obtaining a fractional solution for the continuous relaxation of a given instance.

**Overview** Our algorithm consists of two main phases to which we refer as profit enumeration and the randomized procedure. The randomized procedure returns a feasible solution for its input instance, whose expected value is at least $(1 - \Theta(\varepsilon))(1 - e^{-1})$ times the optimal solution, minus $\Theta(M_I)$, where $M_I$ is the maximal value of a single element in this instance. Hence, to guarantee a constant approximation ratio (by expectation), the profit enumeration phase guesses (by enumeration) a constant number of elements of highest value in some optimal solution; then the algorithm proceeds to the randomized procedure taking the value residual problem with respect to the guessed subset. Since the maximal value of a single element in the value residual problem is bounded, we obtain the desired approximation ratio.

[3]The $o(1)$ factor can be eliminated.

The randomized procedure uses randomized rounding in order to attain an integral solution from a fractional solution returned by the algorithm of [16]. However, simple randomized rounding may not guarantee a feasible solution, as some of the linear constraints may be violated. This is handled by the following steps. First, the algorithm enumerates on the big elements in an optimal solution: this enables to bound the variance of the cost in each dimension, and the event of discarding an infeasible solution occurs with small probability. Second, we apply a *fixing procedure*, in which a *nearly* feasible solution is converted to a feasible solution, with small harm to the objective function.

## 2.2 Profit Enumeration

In section 2.3 we present algorithm MLC_RR$_{\varepsilon,d}(I)$. Given an instance $I$ of MLC and some $\varepsilon > 0$, MLC_RR$_{\varepsilon,d}(I)$ returns a feasible solution for $I$ whose expected value is at least $(1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O} - d\varepsilon^{-3}M_I$, where

$$(2.1) \qquad M_I = \max_{i \in U} f(\{i\})$$

is the maximal value of any element in $I$, and $\mathcal{O}$ is the value of the optimal solution.

We use this algorithm as a procedure in the following.

**Approximation Algorithm for MLC ($\mathcal{A}_{MLC}$)**

1. For any $T \subseteq U$ such that $|T| \leq \lceil ed \cdot \varepsilon^{-3} \rceil$:
   (a) $S \leftarrow$ MLC_$RR_{\varepsilon,d}(I_T)$, where $I_T$ is the value residual problem with respect to $T$.
   (b) if $f(S \cup T) > f(\mathcal{D})$ then set $\mathcal{D} = S \cup T$.
2. Return $\mathcal{D}$

THEOREM 2.1. *Algorithm $\mathcal{A}_{MLC}$ runs in polynomial time and returns a feasible solution for the input instance I, with expected approximation ratio of* $(1 - \Theta(\varepsilon))(1 - e^{-1})$.

The above theorem implies that, for any $\hat{\varepsilon} > 0$, with a proper choice of $\varepsilon$, $\mathcal{A}_{MLC}$ is a polynomial time $(1 - \hat{\varepsilon})(1 - e^{-1})$-approximation algorithm for MLC.

*Proof.* Let $\mathcal{O} = \{i_1, \ldots, i_k\}$ be an optimal solution for $I$ (we use $\mathcal{O}$ to denote both an optimal sub-collection of elements and the optimal value). Let $h = \lceil ed \cdot \varepsilon^{-3} \rceil$, and $K_\ell = \{i_1, \ldots, i_\ell\}$ (for any $\ell \geq 1$), and assume that the elements are ordered by their residual profits, i.e., $i_\ell = \text{argmax}_{i \in OPT \setminus K_{\ell-1}} f_{K_{\ell-1}}(\{i\})$.

Clearly, if there are less than $h$ elements in $\mathcal{O}$, then these elements are considered in some iteration of $\mathcal{A}_{MLC}$, and the algorithm finds an optimal solution; otherwise, consider the iteration in which $T = K_h$. For any $j > h$, $f_{K_{h-1}}(\{j\}) \leq f_{K_{h-1}}(\{h\}) \leq \frac{f(K_h)}{|K_h|}$. Hence, the elements $i_{h+1}, \ldots, i_k$ belong to the value residual problem with respect to $T = K_h$, and the optimal solution of the residual problem is $f_T(\mathcal{O} \setminus K_h) = f_T(\mathcal{O})$.

For some $\alpha \in [0,1]$, let $f(T) = \alpha \cdot \mathcal{O}$. Then the optimal solution for the residual problem is $(1-\alpha)\mathcal{O}$. Hence, by Theorem 2.2 (see in Section 2.3), the expected profit of MLC_RR$_{\varepsilon,d}(I_T)$ is at least

$$(1 - c\varepsilon)(1 - e^{-1})(1 - \alpha) \cdot \mathcal{O} - d\varepsilon^{-3} M_{I_T},$$

where $c > 0$ is some constant, and $M_{I_T}$ is defined in (2.1). By the definition of the residual problem, we get that $M_{I_T} \leq \frac{f(T)}{|T|} \leq \frac{e^{-1}\varepsilon^3}{d} \cdot \alpha\mathcal{O}$; thus, the expected profit from the solution is at least

$$\alpha\mathcal{O} + (1 - c\varepsilon)(1 - e^{-1})(1 - \alpha)\mathcal{O} - d\varepsilon^{-3} \cdot \frac{\alpha\mathcal{O}}{h} \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O}.$$

The expected profit of the returned solution is at least the expected profit in any iteration of the algorithm. This yields the desired approximation ratio.

For the running time of the algorithm we note that, for fixed values of $d \geq 1$ and $\varepsilon > 0$, the number of iterations of the loop is polynomial in the number of sets, and each iteration takes a polynomial number of steps. □

## 2.3 The Randomized Procedure

For the randomized procedure, we use the following algorithm which is parametrized by $\varepsilon$ and $d$ and accepts an input $I$:

**Rounding Procedure for MLC** (MLC_RR$_{\varepsilon,d}(I)$)

1. Enumerate on all possible sub-collections of big elements which yield feasible solutions; denote the chosen sub-collection by $T$, and let $T_r \subseteq T$ be the sub-collection of elements in $T$ which are big in the $r$-th dimension, $r = 1, \ldots, d$. Denote by $I_T$ the cost residual problem with respect to $T$.
   (a) Find $\bar{x}$ in the polytope of $I_T$ such that $F_T(\bar{x})$ is at least $(1-e^{-1}-\varepsilon)$ times the optimal solution of $I_T$.
   (b) Add any small element $i$ to the solution with probability $(1-\varepsilon)x_i$; add any element $i \in T$ to the solution with probability $(1-\varepsilon)$. Denote the selected elements by $\mathcal{D}$.
   (c) For any $1 \leq r \leq d$, let $L_r^g = \sum_{i \in T_r} c_{i,r}$, and $\tilde{L}_r = L_r - L_r^g$.
   (d) If for some $1 \leq r \leq d$ one of the following holds:
      - $\tilde{L}_r > \varepsilon L_r$ and $\sum_{i \in \mathcal{D}} c_{i,r} > L_r$
      - $\tilde{L}_r \leq \varepsilon L_r$ and $\sum_{i \in \mathcal{D} \setminus T_r} c_{i,r} > \varepsilon L_r + \tilde{L}_r$

      then select $\mathcal{D} = \emptyset$, **else**
   (e) For any dimension $1 \leq r \leq d$ such that $\tilde{L}_r \leq \varepsilon L_r$, remove from $\mathcal{D}$ elements in $T_r$ until $\sum_{i \in \mathcal{D}} c_{i,r} \leq L_r$.
   (f) If $f(\mathcal{D})$ is larger than the value of the current best solution, then set $\mathcal{D}$ to be the current best solution
2. Return the best solution.

We now analyze algorithm $\mathcal{A}_{MLC}$. For an instance $I$ of MLC, let $\mathcal{O}$ be an optimal solution ($\mathcal{O}$ is used both as the selected set of elements, and as the value of the solution).

THEOREM 2.2. *Given an input $I$, algorithm* MLC_RR$_{\varepsilon,d}(I)$ *returns a feasible subset of elements $S$ such that*

$$E[f(S)] \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O} - d\varepsilon^{-3} M_I,$$

*where $M_I$ is defined in (2.1).*

We consider the iteration in which $T$ contains exactly all the big elements in $\mathcal{O}$. To prove Theorem 2.2, we use the next technical lemmas. First, define $W = f(\mathcal{D})$ when $\mathcal{D}$ is considered after stage (1b), then

LEMMA 2.1. *$E[W] \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O}$.*

*Proof.* Let $D_1$ be the collection of small elements in $\mathcal{D}$, and $D_2 = \mathcal{D} \setminus D_1$ the collection of big elements in $\mathcal{D}$. In Step (1a) we get that $F_T(\bar{x}) \geq (1 - e^{-1} - \varepsilon) f_T(\mathcal{O})$ (the optimal solution for $I_T$ is $f_T(\mathcal{O})$, by the selection of $\mathcal{O} \setminus T$). Hence, due to the convexity of $F$ (see [16], we have that

$$E[f_T(D_1)] = F((1-\varepsilon)\bar{x}) \geq (1-\varepsilon)(1 - e^{-1} - \varepsilon) f_T(\mathcal{O}),$$

and

$$E[f(D_2)] = F((1-\varepsilon)\mathbf{1}_T) \geq (1-\varepsilon)F(\mathbf{1}_T) = (1-\varepsilon)f(T).$$

( $y = \mathbf{1}_T \in \{0,1\}^U$ such that $y_i = 1$ iff $i \in T$). It follows that

$$\begin{aligned} E[W] &= E[f(\mathcal{D})] = E[f(D_1) + f_{D_1}(D_2)] \\ &\geq E[f(D_2)] + E[f_T(D_1)] \\ &\geq (1-\varepsilon)f(T) + (1-\varepsilon)(1-e^{-1}-\varepsilon)f_T(\mathcal{O}) \\ &\geq (1-\Theta(\varepsilon))(1-e^{-1})\mathcal{O}. \end{aligned}$$

□

Lemma 2.1 implies that, after the randomized rounding of stage (1b), the integral solution $\mathcal{D}$ has a high value. In the next lemma we show that the modifications applied to $\mathcal{D}$ in stages (1d) and (1e) may cause only small harm to the expected value of the solution.

We say that a solution is *nearly feasible in dimension r* if it does not satisfy any of the conditions in (1d), and *nearly feasible*, if it is nearly feasible in each dimension. Let $F$ ($F_r$) be an indicator for the feasibility of $\mathcal{D}$ (feasibility of $\mathcal{D}$ in dimension $r$) after stage (1b).

Lemma 2.2. *$Pr(F = 0) \leq d\varepsilon$.*

*Proof.* For some $1 \leq r \leq d$, let $Z_{r,1}$ be the cost of $\mathcal{D} \cap T_r$ in dimension $r$, and let $Z_{r,2}$ be the cost of $\mathcal{D} \setminus T_r$ in dimension $r$. Clearly, $Z_{r,1} \leq L_r^g (\leq L_r)$. Let $X_i$ be an indicator random variable for the selection of the element $i$ (note that the $X_i$'s are independent). Let $Small(r)$ be the collection of all the elements which are not big in dimension $r$, i.e., for any $i \in Small(r)$, $c_{i,r} < \varepsilon^4 L_r$. Then, $Z_{r,2} = \sum_{i \in Small(r)} X_i c_{i,r}$. It follows that $E[Z_{r,2}] = \sum_{i \in Small(r)} c_{i,r} E[X_i] \leq (1-\varepsilon)\tilde{L}_r$, and

$$Var[Z_{r,2}] \leq \sum_{i \in Small(r)} E[X_i] c_{i,r} \cdot \varepsilon^4 L_r \leq \varepsilon^4 L_r \tilde{L}_r.$$

Recall that by the Chebyshev-Cantelli bound, for any $t > 0$,

$$Pr(Z_{r,2} - E[Z_{r,2}] \geq t) \leq \frac{Var[Z_{r,2}]}{Var[Z_{r,2}] + t^2}.$$

Thus, if $\tilde{L}_r > \varepsilon L_r$, using the Chebyshev-Cantelli inequality, we have

$$Pr(F_r = 0) Pr(Z_{r,2} - E[Z_{r,2}] > \varepsilon \tilde{L}_r) \leq \frac{\varepsilon^4 L_r \tilde{L}_r}{\varepsilon^2 \tilde{L}_r^2} \leq \varepsilon;$$

else $\tilde{L}_r \leq \varepsilon L_r$. Similarly,

$$\begin{aligned} Pr(F_r = 0) &\leq Pr(Z_{r,2} - E[Z_{r,2}] > \varepsilon L_r) \\ &\leq \frac{\varepsilon^4 L_r \tilde{L}_r}{\varepsilon^2 {L_r}^2} \leq \varepsilon^3. \end{aligned}$$

By the union bound, we get that $Pr(F = 0) \leq d\varepsilon$. □

For any dimension $r$, let $R_r = \frac{\sum_{i \in \mathcal{D}} c_{i,r}}{L_r}$ and define $R = \max_r R_r$, where $\mathcal{D}$ is considered after stage (1b).

Lemma 2.3. *For any $\ell > 1$,*

$$Pr(R > \ell) < \frac{d\varepsilon^4}{(\ell-1)^2}.$$

*Proof.* For any dimension $1 \leq r \leq d$, let $Z_{r,1}$, $Z_{r,2}$ be defined as in the proof of Lemma 2.2. By the Chebyshev-Cantelli inequality, we have that

$$\begin{aligned} Pr(R_r > \ell) &= Pr(Z_{r,2} > \ell \cdot L_r - L_r^g) \\ &\leq Pr(Z_{r,2} - E[Z_{r,2}] > (\ell - 1)L_r) \\ &\leq \frac{\varepsilon^4 L_r \tilde{L}_r}{(\ell-1)^2 {L_r}^2} \leq \frac{\varepsilon^4}{(\ell-1)^2}, \end{aligned}$$

and by the union bound, we get that

$$Pr(R > \ell) \leq \frac{d\varepsilon^4}{(\ell-1)^2}.$$

□

Lemma 2.4. *For any integer $\ell > 1$, if $R \leq \ell$ then*

$$f(\mathcal{D}) \leq 2d\ell \cdot \mathcal{O}.$$

*Proof sketch.* The set $\mathcal{D}$ can be partitioned to $2d\ell$ sets $\mathcal{D}_1, \ldots \mathcal{D}_{2d\ell}$ such that each of this sets is a feasible solution. Hence, $f(\mathcal{D}_i) \leq \mathcal{O}$, and so $f(\mathcal{D}) \leq f(\mathcal{D}_1) + \ldots + f(\mathcal{D}_{2d\ell}) \leq 2d\ell f(\mathcal{O})$. □

Let $W' = f(\mathcal{D})$ when $\mathcal{D}$ is considered after stage (1d).

Lemma 2.5. *$E[W'] \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O}$.*

*Proof.* By Lemmas 2.2 and 2.3, it holds that

$$\begin{aligned} E[W] &= E\left[W | F = 1\right] \cdot Pr(F = 1) + \\ &\quad E\left[W | F = 0 \wedge R < 2\right] \cdot Pr(F = 0 \wedge (R < 2)) \\ &\quad + \sum_{\ell=1}^{\infty} E\left[W | F = 0 \wedge (2^\ell \leq R \leq 2^{\ell+1})\right] \\ &\qquad \cdot Pr(F = 0 \wedge (2^\ell \leq R \leq 2^{\ell+1})) \\ &\leq E[W | F = 1] \cdot Pr(F = 1) + 4d^2 \hat{\varepsilon} \cdot \mathcal{O} \\ &\quad + d^2 \hat{\varepsilon}^4 \cdot \mathcal{O} \cdot \sum_{\ell=1}^{\infty} \frac{2^{\ell+2}}{(2^{\ell-1})^2}. \end{aligned}$$

Since the last summation is a constant, using Lemma 2.1, we have that:

$$E[W | F = 1] \cdot Pr(F = 1) \geq (1 - c\hat{\varepsilon})(1 - e^{-1})\mathcal{O},$$

where $c$ is some constant. Also, since $W' = W$ if $F = 1$ and $W' = 0$; otherwise, we have that

$$E[W'] = E[W|F] \cdot Pr(F) \geq (1 - c\hat{\varepsilon})(1 - e^{-1})\mathcal{O}.$$

□

LEMMA 2.6. *Let $P = f(\mathcal{D})$ when $\mathcal{D}$ is considered after stage (1f). Then $\mathcal{D}$ is a feasible solution, and*

$$E[P] \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O} - d\varepsilon^{-3} \cdot M_I.$$

*Proof.* In stage (1e), for each dimension $1 \leq r \leq d$, if $\tilde{L}_r > \varepsilon L_r$ then no elements are removed from the solution (and, clearly, the solution is feasible in this dimension). If $\tilde{L}_r \leq \varepsilon L_r$ then, if all big elements in the $r$-th dimension are removed, the solution becomes feasible in this dimension, since

$$\sum_{i \in \mathcal{D} \setminus T_r} c_{i,r} \leq \tilde{L}_r + \varepsilon L_r \leq 2\varepsilon L_r \leq L_r$$

(for $\varepsilon < 1/2$). This implies that it is possible to convert the solution to a feasible solution in the $r$-th dimension by removing only elements which are big in this dimension. At most $\varepsilon^{-3}$ elements need to be removed due to each dimension $r$ (since $c_{i,r} \geq \varepsilon^4 L_r$ when $i$ is big in the $r$th dimension). Hence, in stage (1e) at most $d\varepsilon^{-3}$ elements are removed. Then the expected value of the solution after this stage satisfies $E[P] \geq E[W'] - d\varepsilon^{-3} M_I$ (since the profit is a non-decreasing submodular function) and, by Lemma 2.5,

$$E[P] \geq (1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O} - d\varepsilon^{-3} \cdot M_I.$$

□

*Proof of Theorem 2.2.* Since any non-feasible solution is converted to a feasible one, the algorithm returns a feasible solution.

By Lemma 2.6, the expected value of the returned solution is at least $(1 - \Theta(\varepsilon))(1 - e^{-1})\mathcal{O} - d\varepsilon^{-3} \cdot M_I$. For the running time of the algorithm, we note that each iteration of the loop runs in polynomial time; the number of iterations of the main loop is also polynomial, as the number of sets in $T$ is bounded by $d\varepsilon^{-4}$, which is a constant for fixed values of $d \geq 1$, $\varepsilon > 0$. □

## 3 Approximation Algorithm for MCP

The MCMP problem in single dimension can be formulated as follows. Given is a ground set $A = \{a_1, ..., a_n\}$, where each element $a_j$ has a weight $w_j \geq 0$ and size $s_j \geq 0$. Also, given are a size limit $B$, a collection of subsets $S = \{S_1, ..., S_m\}$, and an integer $k > 1$.

Let $s(\mathcal{E}) = \sum_{a_j \in \mathcal{E}} s_j$ for all $\mathcal{E} \subseteq A$, and $w(\mathcal{E}) = \sum_{a_j \in \mathcal{E}} w_j$. The goal is to select a sub-collection of sets $S'$, such that $|S'| \leq k$, and a set of elements $\mathcal{E} \subseteq \bigcup_{S_i \in S'} S_i$, such that $s(\mathcal{E}) \leq B$ and $w(\mathcal{E})$ is maximized.

Let $\mathcal{O}$ be an optimal solution for MCP (we use $\mathcal{O}$ also as the weight of the solution). Our approximation algorithm for MCP combines an enumeration stage, which involves guessing the $\ell = 3$ elements with highest weights in $\mathcal{O}$ and sets that cover them, with maximization of a submodular function.

More specifically, we arbitrarily associate each element $a_j$ in $\mathcal{O}$ with a set $S_i$ in $\mathcal{O}$ which contains $a_j$; we then consider them as a pair $(a_j, S_i)$. The first stage of our algorithm is to guess $T$, a collection of $\ell$ pairs $(a_j, S_i)$, such that $a_j \in S_i$. The elements in $T$ are the $\ell$ elements with highest weights in $\mathcal{O}$. Let $T_{\mathcal{E}}$ be the collection of elements in $T$, and let $T_S$ be the collection of sets in $T$. Also, let $k' = k - |T_S|$ and $w_T = \min_{a_j \in T_{\mathcal{E}}} w_j$. We denote by $\mathcal{O}'$ the weight of the solution $\mathcal{O}$ excluding the elements in $T_{\mathcal{E}}$; then, $w(T_{\mathcal{E}}) = \mathcal{O} - \mathcal{O}'$.

We use our guess of $T$ to define a non-decreasing submodular set-function over $S \setminus T_S$. Let $B' = B - s(T_{\mathcal{E}})$. We first define a function $g : 2^A \to \mathbb{R}$:

$$\begin{array}{lll} g(\mathcal{E}) = & \max & \displaystyle\sum_{j=1}^{n} x_j w_j \\ & \text{subject to:} & 0 \leq x_j \leq 1 \qquad \forall\, a_j \in \mathcal{E} \\ & & x_j = 0 \qquad \forall\, a_j \notin \mathcal{E} \\ & & \displaystyle\sum_{j=1}^{n} x_j s_j \leq B' \end{array}$$

Note that while $g$ is formulated as a linear program, given a collection of elements $\mathcal{E}$, the value of $g(\mathcal{E})$ can be easily evaluated by a simple greedy algorithm, and the vector $\bar{x}$ for which the value of $g(\mathcal{E})$ is attained has a single fractional entry.

For any $S' \subseteq S \setminus T_S$ define

$$C(S') = \{a_j |\ a_j \in \textstyle\bigcup_{S_i \in S' \cup T_S} S_i,\ a_j \notin T_{\mathcal{E}},\ w_j \leq w_T\}.$$

We use $g$ and $C(S')$ to define $f : 2^{S \setminus T_S} \to \mathbb{R}$ by $f(S') = g(C(S'))$.

Consider the problem

$$\text{(3.2)} \qquad \max f(S') \ \text{ subject to: } \ |S'| \leq k'.$$

By taking $S'$ to be all the sets in $\mathcal{O}$ excluding $T_S$, we get that $|S'| \leq k'$ and $f(S') \geq \mathcal{O}'$. This gives a lower bound for the value of the problem. To find a collection of subsets $S'$ such that $f(S')$ is an approximation for the problem (3.2), we use the following property of $f$:

LEMMA 3.1. *The function $f$ is a non-decreasing submodular set function.*

This means that we can find a $(1-e^{-1})$ approximation for the problem (3.2) by using a greedy algorithm [11]. Let $S'$ be the collections of subsets obtained by this algorithm. Since it is a $(1-e^{-1})$-approximation for (3.2), we have that $g(C(S')) = f(S') \geq (1-e^{-1})\mathcal{O}'$. Consider the vector $\bar{x}$ which maximizes $g(C(S'))$, such that $\bar{x}$ has at most one fractional entry. (As mention above, such $\bar{x}$ can be found using a simple greedy algorithm.) Consider the collection of elements $\mathcal{C} = \{a_j | x_j = 1\}$. Since there is at most one fractional entry in $\bar{x}$, by the definition of $C(S')$ we have that $w(\mathcal{C}) \geq g(C(S')) - w_T$. Now consider the collection of sets $S' \cup T_S$, along with the elements $\mathcal{C} \cup T_{\mathcal{E}}$. This is a feasible solution for MCP whose total weight is at least

$$w(T_{\mathcal{E}}) + g(C(S')) - w_T \geq (1-\frac{1}{\ell})w(T_{\mathcal{E}}) + (1-e^{-1})\mathcal{O}'$$
$$= (1-\frac{1}{\ell})(\mathcal{O}-\mathcal{O}') + (1-e^{-1})\mathcal{O}' \geq (1-e^{-1})\mathcal{O}.$$

The last inequality follows from the fact that $\ell = 3$, therefore $(1-\frac{1}{l}) \geq 1-e^{-1}$. We now summarize the steps of our approximation algorithm.

**Approximation Algorithm For MCP**

1. Enumerate on all the possible sets $T$ of pairs $(a_j, S_i)$, such that $a_j \in S_i$ and $|T| \leq \ell$:
   (a) Find a $(1-e^{-1})$-approximation $S'$ for the problem (3.2), using the greedy algorithm [11].
   (b) Let $\bar{x}$ be the vector that maximizes $g(C(S'))$, such that $\bar{x}$ has at most one fractional entry. Define $\mathcal{C} = \{a_j | x_j = 1\}$
   (c) Consider the collection of sets $S' \cup T_S$, along with the elements $\mathcal{C} \cup T_{\mathcal{E}}$. If the weight of this solution is higher than the best solution found so far, select it as the best solution.
2. Return the best solution found.

By the above discussion, we have

THEOREM 3.1. *The approximation algorithm for MCP achieves a ratio of $1-e^{-1}$ to the optimal and has a polynomial running time.*

The result in this section can be easily extended to solve a generalization of MCP where each set has a cost $c_i$, and there is a budget $L$ for the sets, by using an algorithm of [15]. In contrast, there is no immediate extension of the above result to MCMP. A main obstacle is the fact that when attempting to define a function $g$ (and accordingly $f$) that involves more than a single linear constraint, the resulting function is not submodular.

## 4 A Randomized Approximation Algorithm for MCMP

The problem of maximum coverage with multiple packing constraint is the following variant of the maximum coverage problem. Given is a collection of sets $S = \{S_1, ..., S_m\}$ over a ground set $A = \{a_1, ..., a_n\}$, where each element $a_j$ has a weight $w_j \geq 0$ and a $d$-dimensional size vector $\bar{s}_j = (s_{j,1}, \ldots, s_{j,d})$, such that $s_{j,r} \geq 0$ for all $1 \leq r \leq d$. Also, given is an integer $k > 1$, and a bin whose capacity is given by the $d$-dim vector $\bar{B} = (B_1, \ldots, B_d)$. A collection of elements $\mathcal{E}$ is *feasible* if for any $1 \leq r \leq d$, $\sum_{a_j \in \mathcal{E}} s_{i,r} \leq B_r$ ; the weight of $\mathcal{E}$ is $w(\mathcal{E}) = \sum_{a_j \in \mathcal{E}} w_j$. The goal is to select a sub-collection of sets $S' \subseteq S$ of size at most $k$ and a feasible collection of elements $\mathcal{E} \subseteq A$, such that each element in $\mathcal{E}$ is an element in some $S_i \in S'$ and $w(\mathcal{E})$ is maximized.

An important observation when attempting to solve this problem is that given the selected sub-collection of sets $S'$, choosing the sub-set of elements $\mathcal{E}$ (which are covered by $S'$) yields an instance of the classic *Multidimensional Knapsack Problem* (MKP) It is well known that MKP admits a PTAS [6], and that the existence of an FPTAS for the problem would imply that $P = NP$ (see, e.g., [8]). Our algorithm makes use of the two main building blocks of the PTAS for MKP as presented in [8], namely, an exhaustive enumeration stage, combined with certain properties of the linear programing relaxation of the problem.

Let $\mathcal{O}$ be an optimal solution for the given instance for MCMP. We arbitrarily associated each element $a_j$ selected in $\mathcal{O}$ with some selected subset $S_i$ in $\mathcal{O}$ such that $a_j \in S_i$. For the use of our algorithm, we guess a collection $T$ of $\ell$ pairs $(a_j, S_i)$ of an element $a_j$ and a set $S_i$ with which it is associated, such that the collection of elements in $T$ forms the $\ell$ elements with highest weight in $\mathcal{O}$. Let $T_{\mathcal{E}}$ be the collection of elements in $T$ and $T_S$ be the collection of sets in $T$. Also, let $w_T = \min_{a_j \in T_{\mathcal{E}}} w_j$.

After guessing the pairs in $T$ and taking them as the initial solution for the problem, we use the following notation, which reflects the problem that now needs to be solved. Define the capacity vector $\bar{B}' = (B'_1, \ldots, B'_d)$ where $B'_r = B_r - \sum_{a_j \in T_{\mathcal{E}}} s_{j,r}$ for $1 \leq r \leq d$. We reduce the collection of elements to

$$A' = \{a_j \in A \setminus T_{\mathcal{E}} | w_j \leq w_T \ \text{ and } \ \bar{s}_j \leq \bar{B}'\}$$

($A'$ consists of all the elements whose weight is not greater than the smallest weight of an element in $T$, and fit into the new capacity-vector). Define the subsets to be $S'_i = S_i \cap A'$. Also, let $\mathcal{O}' = \mathcal{O} - \sum_{a_j \in T_{\mathcal{E}}} w_j$ be the total weight in the optimal solution from elements not in $T$.

We define the size of a subset $S'_i$ to be the total size of elements associated with $S'_i$ (i.e., the elements associated with $S_i$, excluding elements in $T_{\mathcal{E}}$), and denote it by $\hat{\bar{s}}_i = (\hat{s}_{i,1}, \ldots, \hat{s}_{i,d})$. We say the a subset is *big in dimension* $r$ if $\hat{s}_{i,r} > \varepsilon^4 B'_r$, and *small in dimension* $r$ otherwise. Since there are up to $\varepsilon^{-4}$ subsets that are big in dimension $r$ in $\mathcal{O}$, we can guess which sets are big in each dimension in the solution $\mathcal{O}$. Let $G_r$ be the collection of big sets in dimension $r$ in our guess. Also, let $x_i \in \{0,1\}$ be an indicator for the selection of $S'_i$, $1 \leq i \leq m$; $y_{i,j} \in \{0,1\}$ indicates whether $a_j \in A'$ is associated with $S'_i$, $1 \leq i \leq m$. Using the guess of $T$ and our guess of the *big* sets, we define the following linear programing relaxation for the problem.

(4.3)

$$\begin{array}{ll}
\text{maximize} & \displaystyle\sum_{i=1}^{m} \sum_{j | a_j \in S'_i} y_{i,j} w_j \\
\text{subject to:} & \\
\forall i,j \text{ s.t. } a_j \in S'_i : & y_{i,j} \leq x_i \\
\forall a_j \in A' : & \displaystyle\sum_{i | a_j \in S'_i} y_{i,j} \leq 1 \\
 & \displaystyle\sum_{i=1}^{m} x_i \leq k \\
\forall 1 \leq r \leq d : & \displaystyle\sum_{i=1}^{m} \sum_{j | a_j \in S'_i} y_{i,j} s_{j,r} \leq B'_r \\
\forall S_i \in T_S : & x_i = 1 \\
\forall 1 \leq r \leq d \wedge S'_i \in G_r : & x_i = 1, \displaystyle\sum_{a_j \in S'_i} y_{i,j} s_{j,r} \geq \varepsilon^4 B'_r \\
\forall 1 \leq r \leq d \wedge S'_i \notin G_r : & \displaystyle\sum_{a_j \in S'_i} y_{i,j} s_{j,r} \leq \varepsilon^4 B'_r x_i
\end{array}$$

It is important to note that, given the current guess of $T$ and $G_r$ for every $1 \leq r \leq d$, the value of the optimal solution for (4.3) is at least $\mathcal{O}'$. The guessed sets $G_r$ were involved in the last two constraints of the system. All sets in $G_r$ were added to the solution ($x_i = 1$), and are 'forced' to be *big*. In contrast, sets which are not in $G_r$ have to satisfy the constraint $\sum_{a_j \in S'_i} y_{i,j} s_{j,r} \leq \varepsilon^4 B'_r x_i$. Therefore, these sets remain small even after scaling the values of $y_{i,j}$ by $x_i^{-1}$.

The solution of the linear program (4.3) is used to randomly determine the sets selected for our solution. For any set $S_i$, $1 \leq i \leq m$, if $S_i \in T_S$ then add it to the solution. Otherwise, add $S_i$ to the solution with probability $(1-\varepsilon)x_i$. If the resulting solution $\mathcal{D}$ contains more than $k$ subsets, then return an empty solution; otherwise, define $\mathcal{C} = \bigcup_{S_i \in \mathcal{D}} S'_i$ and solve the following linear program:

$$(4.4) \quad \begin{array}{lll}
\text{maximize:} & & \displaystyle\sum_{a_j \in \mathcal{C}} y_j w_j \\
\text{subject to:} & \forall \ 1 \leq r \leq d : & \displaystyle\sum_{a_j \in \mathcal{C}} y_j s_{j,r} \leq B'_r \\
 & \forall \ a_j \in \mathcal{C} : & 0 \leq y_j \leq 1
\end{array}$$

A **basic** solution for the above linear program has at most $d$ fractional entries. Let $\mathcal{A}$ be the collection of elements $a_j \in \mathcal{C}$ for which $y_j = 1$; then, clearly, $\mathcal{A} \cup T_{\mathcal{E}}$ along with the collection of subsets $\mathcal{D}$ forms a feasible solution for the problem.

We now summarize the steps of our algorithm, which gets as input the parameters $\ell \geq 1$ and $\varepsilon > 0$.

**Approximation Algorithm for MCMP**

1. If $k \leq \varepsilon^{-3} + \ell$ enumerate on the subsets in the optimal solution, and run the PTAS for MKP for selecting the elements. (this guaranties an approximation ratio of $(1-\varepsilon)$)
2. For each collection $T$ of pairs $(a_j, S_i)$ (where $a_j \in S_i$) of size at most $\ell$, and any $G_r \subseteq \{1, \ldots, m\}$ of size at most $\varepsilon^{-4}$ do the following:
   (a) Solve (4.3) with respect to $T$,$G_r$. Let $\bar{x} = (x_1, \ldots, x_m)$ be the (partial) solution.
   (b) Initially, let $\mathcal{D} = \emptyset$. For any $S_i \in S$, if $S_i \in T_S$ add $S_i$ to $\mathcal{D}$; otherwise, add $S_i$ to $\mathcal{D}$ with probability $(1-\varepsilon)x_i$.
   (c) If the number of sets in $\mathcal{D}$ is greater than $k$, continue to the next iteration of the loop
   (d) Solve the linear program (4.4). Let $\bar{y}$ be the solution. Set $\mathcal{A}$ to be all the element $a_j$ such that $y_j = 1$.
   (e) If the weight of elements in $\mathcal{A} \cup T_{\mathcal{E}}$ is greater than the weight of the current best solution, choose $\mathcal{A} \cup T_{\mathcal{E}}$ with $\mathcal{D}$ as the current best solution.
3. Return the best solution found

It is easy to verify that the running time of the algorithm is polynomial (for fixed $\ell$ and $\varepsilon$) since the number of iterations of the main loop is polynomial.

Also, clearly, the solution returned by the algorithm is always feasible. It remains to show that the algorithm admits the desired approximation ratio of $\alpha_f = 1 - (1 - \frac{1}{f})^f > (1 - e^{-1})$, where $f$ is the maximal number of subsets in which a single element appears. For the case where $k \leq \varepsilon^{-3} + \ell$, the claim is trivial. Hence, we assume below that $k > \varepsilon^{-3} + \ell$.

To show the approximation ratio of $\alpha_f$, we refer to the iteration in which we use the correct guess for $T$ and $G_r$. We define a slightly more complex randomized process. For any $S_i$ such that $S_i \notin T_S$, let $X_i$ be an indicator random variable, where $X_i = 1$ if $S_i \in \mathcal{D}$ and $X_i = 0$ otherwise. For any $S_i \in T_S$, let $X_i = 1$ with probability $(1 - \varepsilon)$ and $X_i = 0$ with probability $\varepsilon$. This defines the distribution of $X_i$, $1 \leq i \leq m$: $X_i = 1$ with probability $(1 - \varepsilon)x_i$, and $X_i = 0$ otherwise. We note that the $X_i$'s are independent random variables.

Let $\{y_{i,j}\}$ be the solution obtained for (4.3) in line (2a). The values of $X_i$'s, are used to determine the value of the random variable $Y_j$, for any $a_j \in A'$, namely,

$$Y_j = \min\left\{1, \sum_{i|a_j \in S_i} \frac{y_{i,j}}{x_i} X_i\right\}.$$

Our goal is to show that (a slight modification of) $Y_1, \ldots, Y_n$ forms a solution for (4.4) with high value. Define $y_j = \sum_{i|a_j \in S_i} y_{i,j}$, then the following holds.

LEMMA 4.1. *For any $a_j \in A'$,*

$$E[Y_j] \geq (1 - \varepsilon) \cdot \alpha_f \cdot y_j.$$

We use another random variable, $Y = \sum_{a_j \in A'} Y_j w_j$; $Y$ can be viewed as the value of $\{Y_j\}$ when used as a solution for (4.4). Let $OPT$ be the value of the optimal solution for the linear program (4.3). By Lemma 4.1, we have that

$$\begin{aligned} E[Y] &= E\left[\sum_{j \in A'} Y_j w_j\right] \geq (1 - \varepsilon) \cdot \alpha_f \cdot OPT \\ &\geq (1 - \varepsilon) \cdot \alpha_f \cdot \mathcal{O}'. \end{aligned}$$

Define the size of the set $S_i$ to be $\hat{\bar{s}}_i = (\hat{s}_{i,1}, \ldots, \hat{s}_{i,d})$, where $\hat{s}_{i,r} = \sum_{a_j \in S_i'} \frac{y_{i,j}}{x_i}$. For any dimension $r$, let $B_r^g = \sum_{i \in G_r} \hat{s}_{i,r}$, and $\tilde{B}_r = B_r' - B_r^g$. Also, we use the notation

$$Z_{r,1} = \sum_{i \in G_r} X_i \hat{s}_{i,r}\ ,\ \ Z_{r,2} = \sum_{i=1, i \notin G_r}^{m} X_i \cdot \hat{s}_{i,r},$$

and $Z_r = Z_{r,1} + Z_{r,2}$ (the total size of selected big subsets, not-big subsets, and all subsets in dimension $r$).

We say that the solution (the result of the randomized process) is *nearly feasible in dimension* $r$ $(1 \leq r \leq d)$ if one of the following holds:

1. $\tilde{B}_r > \varepsilon B_r'$ and $Z_r \leq B_r'$
2. $\tilde{B}_r \leq \varepsilon B_r'$ and $Z_{r,2} \leq \varepsilon B_r' + \tilde{B}_r$

We use an indicator random variable $F_r$ such that $F_r = 1$ if the solution is feasible in dimension $r$ and $F_r = 0$ otherwise. Though we cannot bound the probability for $Z_r > B_r'$, we are able to bound the probability for $F_r = 0$.

LEMMA 4.2. *For any dimension $1 \leq r \leq d$,*

$$Pr[F_r = 0] \leq \varepsilon.$$

The above lemma bounds the probability for a small deviation of $Z_r$. Larger deviations can be easily bounded. Let $R_r = \frac{Z_r}{B_r'}$, then

LEMMA 4.3. *For any dimension $1 \leq r \leq d$ and $t > 1$,*

$$Pr(R_r > t) \leq \frac{\varepsilon^4}{(t-1)^2}.$$

Next, we bound the probability that more than $k$ sets are selected for the solution. We use the assumption that $k > \varepsilon^{-3} + \ell$.

LEMMA 4.4. *For any $t > 1$,*

$$Pr(||\mathcal{D}|| > t \cdot k) \leq \frac{\varepsilon}{t^2}.$$

Let $R = \max\{\max_r R_r, \frac{|\mathcal{D}|}{k}\}$. The following claim is a direct conclusion from the two last lemmas, by applying the union bound.

CLAIM 4.1. *For any $t > 1$,*

$$Pr(R > t) \leq \frac{d\varepsilon}{(t-1)^2} + \frac{\varepsilon}{t^2}.$$

Let $F$ be a random variable such that $F = 1$ if $|\mathcal{D}| \leq k$ and the solution is nearly feasible in dimension $r$, for any $1 \leq r \leq d$, and $F = 0$ otherwise. The next claim also follows from the previous lemmas, using union bound.

CLAIM 4.2. $Pr[F = 0] \leq (d + 1) \cdot \varepsilon$

Now, we bound the value of $Y$ as a function of $R$.

LEMMA 4.5. *For any integer $t > 1$, if $R \leq t$ then $Y \leq t \cdot c_d \cdot \mathcal{O}'$, where $c_d$ is constant for fixed $d$.*

Combining the results of the above lemmas, we obtain the following.

LEMMA 4.6. *For some $c''_d$,*

$$E[Y|F=1]Pr[F=1] \geq (1 - c''_d\varepsilon) \cdot \alpha_f \cdot \mathcal{O}'.$$

Let $W$ to be $(1-\varepsilon)Y$ if $F=1$ and $W=0$ otherwise ($F=1$ if the solution is nearly feasible in every dimension and $|\mathcal{D}| \leq k$). The value of $W$ is a lower bound for the value of the solution for the linear program in line (2d) (in case this line was not reached by the algorithm, we consider its value as zero). This follows from the fact that we can consider the values of $(1-\varepsilon)Y_j$ as solutions for the linear program, and in case they form a nearly feasible solution, the scaling by $(1-\varepsilon)$ makes them feasible. The requirement that $|\mathcal{D}| \leq k$ guarantees that the linear program would be solved. By Lemma 4.6, we get that

$$E[W] \geq (1-\varepsilon) \cdot (1 - c''_d\varepsilon) \cdot \alpha_f \cdot \mathcal{O}'.$$

Finally, we consider $Q$ as the weight of the solution considered in line (2e). In case this line is not performed by the algorithm, we take $Q = 0$.

LEMMA 4.7. *Assuming $\ell \geq \varepsilon^{-1}$,*

$$E[Q] \geq (1-\varepsilon) \cdot (1 - c''_d\varepsilon) \cdot \alpha_f \cdot \mathcal{O}'.$$

Since the expected value of the solution returned by the algorithm is at least the expected value of the solution in any iteration, we summarize in the next theorem.

THEOREM 4.1. *For any fixed $d$ and $\hat{\varepsilon} > 0$, by properly setting the values of $\varepsilon$ and $\ell$, the algorithm achieves approximation ratio of $(1-\hat{\varepsilon})\alpha_f$ and runs in polynomial time.*

**Acknowledgments.** We thank Seffi Naor for many helpful discussions. We also thank Chandra Chekuri and Uri Feige for insightful comments and suggestions.

## References

[1] A. Ageev and M. Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. Combinatorial Optimization*, 8(3):307–328, 2004.

[2] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *IPCO*, pages 182–196, 2007.

[3] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. In *APPROX-RANDOM*, pages 72–83, 2004.

[4] U. Feige. A threshold of *ln n* for approximating set cover. *J.of ACM*, 45(4):634–652, 1998.

[5] U. Feige, V.S.Mirrokni, and J. Vondrák. Maximizing non-monotone submodular functions. In *FOCS*, 2007.

[6] A. M. Frieze and M. Clarke. Approximation algorithms for the m-dimensional 0-1 knapsack problem: worst-case and probabilistic analyses. *European J. of Operational Research*, 15(1):100–109, 1984.

[7] T. Fujito. Approximation algorithms for submodular set cover with applications. *IEICE Trans. Inf. and Systems*, E83-D(3), 2000.

[8] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 1 edition, October 2004.

[9] S. Khuller, A. Moss, and J. Naor. The budgeted maximum coverage problem. *Inf. Process. Letters*, 70(1):39–45, 1999.

[10] A. Kulik, H. Shachnai, and T. Tamir. Maximizing submodular set functions subject to multiple linear constraints. full version. `http://www.cs.technion.ac.il/~hadas/PUB/max_submodular.pdf`.

[11] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[12] A. Schriejver. *Combinatorial Optimization - polyhedra and efficiency*. Springer Verlag - Berlin Heidelberg, 2003.

[13] H. Shachnai and T. Tamir. Polynomial time approximation schemes for class-constrained packing problems. *J. of Scheduling*, 4(6):313–338, 2001.

[14] A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *44th Symp. on Foundations of Computer Science (FOCS)*, pages 588–597, 2001.

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

[16] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.