FPT Approximation Schemes for Maximizing Submodular Functions

Piotr Skowron

University of Oxford, Oxford, UK
piotr.skowron@cs.ox.ac.uk

Abstract. We investigate the existence of approximation algorithms for maximization of submodular functions, that run in fixed parameter tractable (FPT) time. Given a non-decreasing submodular set function \( v: 2^X \rightarrow \mathbb{R} \) the goal is to select a subset \( S \) of \( K \) elements from \( X \) such that \( v(S) \) is maximized. We identify two properties of set functions, referred to as \( p \)-separability properties, and we argue that many real-life problems can be expressed as maximization of submodular, \( p \)-separable functions, with low values of the parameter \( p \). We present FPT approximation schemes for the minimization and maximization variants of the problem, for several parameters that depend on characteristics of the optimized set function, such as \( p \) and \( K \). We confirm that our algorithms are applicable to a broad class of problems, in particular to problems from computational social choice, such as item selection or winner determination under several multiwinner election systems.

1 Introduction

We study (exponential-time) approximation algorithms for maximizing non-decreasing submodular set functions. A set function \( v: 2^X \rightarrow \mathbb{R} \) is submodular if for each two subsets \( A \subseteq B \subset X \) and each element \( x \in X \setminus B \) it holds that \( v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B) \); \( v \) is non-decreasing if for each two subsets \( A \subseteq B \subset X \) it holds that \( v(A) \leq v(B) \). Our goal is to select a subset \( S \) of \( K \) elements from \( X \) such that the value \( v(S) \) is maximal.

Maximization of non-decreasing submodular functions is a very general problem that is extensively used in various research areas, from recommendation systems [21,28], through voting theory [21,29], image engineering [12,13,25], information retrieval [19,34], network design [15,16], clustering [22], speech recognition [20], to sparse methods [1,6]. Algorithms for maximization of non-decreasing submodular functions are applicable to other general problems of fundamental significance, such as the MaxCover problem [4,27]. The universal relevance of the problem implies that the existence of good (approximation) algorithms for it is highly desired.

Indeed, the problem has already received a considerable amount of attention in the scientific community. For instance, it is known that the greedy algorithm, i.e., the algorithm that starts with the empty set and in each of \( K \) consecutive steps adds to the partial solution such an element from \( X \) that increases the value
of the optimized function most, is an \((1 - 1/e)\)-approximation algorithm for maximization of non-decreasing submodular functions [23]. The same approximation ratio can be achieved for the distributed [17] and online [30] variants of the problem. Algorithms for maximizing non-monotone submodular functions have been studied by Feige et al. [9], and the approximability of the problem with additional constraints has been investigated by Calinescu et al. [2], Sviridenko [31], Lee et al. [18], and Vondrák et al. [33]. Iwata et al. [11] have provided algorithmic view on minimizing submodular functions. For the survey on maximization of submodular functions we refer the reader to the work of Krause and Golovin [14].

Unfortunately, the approximation guarantees of the greedy algorithm cannot be improved without compromising the efficiency of computation. For example, the \textsc{MaxCover} problem can be expressed as maximization of a non-decreasing submodular function, yet it is known that under standard complexity assumptions no polynomial-time algorithm can approximate it better than with ratio \((1 - 1/e)\) [8]. Motivated by this fact, and provoked by the desire to obtain better approximation guarantees, we turn our attention to algorithms that run in super-polynomial time. In our studies we follow the approach of parameterized complexity theory and look for algorithms that run in fixed parameter tractable time (in FPT time), for some natural parameters. To the best of our knowledge, FPT approximation of optimizing submodular functions has not been considered in the literature before.

Parameterized complexity theory aims at investigating how the complexity of a problem depends on the size of different parts of input instances, called parameters. An algorithm runs in FPT time for a parameter \(P\) if it solves each instance \(I\) of the problem in time \(O(f(|P|) \cdot \text{poly}(|I|))\), where \(f\) is a computable function. This definition excludes a large class of algorithms, such as the ones with complexity \(O(|I|^{|P|})\). From the point of view of parameterized complexity, FPT is seen as the class of easy problems. Intuitively, the complexity of an FPT algorithm consists of two parts: \(f(|P|)\), which is relatively low for small values of the parameter, and \(\text{poly}(|I|)\) which is relatively low even for larger instances, because of polynomial relation between the computation time and the size of an instance. For details on parameterized complexity theory, we point the reader to appropriate overviews [5,7,10,24].

We identify several parameters that we believe are suitable for a complexity analysis of maximization of non-decreasing submodular functions. Perhaps the most natural parameter to consider is the required size of solutions, \(K\). Our other parameters depend on characteristics of the optimized set function. Specifically, we define a new property of set functions, called \(p\)-separability, and provide evidence that \(p\) is a natural parameter to consider. We do that in Sect. 4, by presenting several examples of real-life computational problems that can be expressed as maximization of submodular \(p\)-separable set functions, where the value of \(p\) is small.

Our main contribution is presentation and analysis of algorithms for the problem. We construct fixed parameter tractable approximation schemes, i.e., collections of algorithms that run in FPT time and that can achieve arbitrarily good approximation ratios. We provide algorithms for two variants of the
problem: in the first variant, referred to as the maximization variant, the goal is to maximize the value \( v(S) \). In the second one, referred to as the minimization variant, the goal is to minimize \( (v(X) - v(S)) \). While these two variants of the problem have the same optimal solutions, they are not equivalent in terms of their approximability. Indeed, if there exists a solution \( S \) with objectively high value, i.e., if \( v(S) \) is close to \( v(X) \), then approximation algorithm for the minimization variant of the problem will be usually superior. For instance, if there exists a solution \( S \) such that \( v(S) = 0.95 \cdot v(X) \), then a 2-approximation algorithm for the minimization variant of the problem is guaranteed to return a solution with the value better than \( 0.9 \cdot v(X) \). On the other hand, a \( 1/2 \)-approximation algorithm for the maximization variant of the problem is allowed to return, in such a case, a solution with value \( 0.475 \cdot v(X) \). Conversely, if the value of an optimal solution is significantly lower than the value of the whole set \( X \), then a good approximation algorithm for the maximization variant of the problem will produce solutions of a better quality.

Our algorithms run in FPT time for the parameter \((K, p)\), where \( K \) is the size of the solution, and \( p \) is the lowest value such that the set function is \( p \)-separable. To address the case of functions which are not \( p \)-separable for any reasonable values \( p \), we define a weaker form of approximability, referred to as approximation of the minimization-or-maximization variant—here, the goal is to find a subset \( S \) that is good in one of the previous two metrics. Such algorithms are also desired as they are guaranteed to find good approximation solutions, provided high quality solutions exist (i.e., if values of the optimal solutions are close to \( v(X) \)). We show that there exists a randomized FPT approximation scheme for minimization-or-maximization variant of the problem for the parameter \((K, \sum_{x \in X} v(\{x\})/v(X))\).

We believe that the consequences of our general results are quite significant. In particular, in Sect. 4, we prove the existence of FPT approximation schemes for some natural problems in the computational social choice, in the matching theory, and in the theoretical computer science.

2 Notation and Definitions

Let \( X \) denote the universe set. We consider a set function \( v : 2^X \to \mathbb{R} \) that is non-negative, i.e., such that for each \( S \subseteq X \) we have \( v(S) \geq 0 \). We say that a function \( v \) is non-decreasing if for each two subsets \( A \subseteq B \subseteq X \) it holds that \( v(B) \geq v(A) \). A set function \( v \) is submodular if for each two subsets \( A \subseteq B \subseteq X \) and each element \( x \in X \setminus B \) it holds that \( v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B) \). There are numerous equivalent conditions characterizing submodular functions—for a survey we refer the reader to the seminal article of Nemhauser et. al. [23]. It is easy to see that if the set function \( v \) is non-decreasing and submodular, then for each two subsets \( A \subseteq B \subseteq X \) and each element \( x \in X \) it holds that \( v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B) \) (here, we do not have to assume that \( x \in X \setminus B \)).

Below, we define a new class of properties of set functions.
Definition 1 (p-separable set function). A submodular set function \( v : 2^X \rightarrow \mathbb{R} \) is:

1. p-superseparable, if for each \( S \subseteq X \) we have:
   \[
   \sum_{x \in X} \left( v(S \cup \{x\}) - v(S) \right) \geq \left( \sum_{x \in X} v(\{x\}) \right) - p \cdot v(S), \tag{1}
   \]

2. p-subseparable, if for each \( S \subseteq X \) we have:
   \[
   \sum_{x \in X} \left( v(S \cup \{x\}) - v(S) \right) \leq p \cdot v(X) - p \cdot v(S). \tag{2}
   \]

For better intuition on the above definitions, we refer the reader to Sect. 4 where we present several examples of natural problems which can be expressed as optimization of separable functions for low values of the parameter \( p \). Indeed, \( p \) can be naturally bounded by \( |X| \): it is easy to see that each monotone and submodular function is \( |X| \)-superseparable and \( |X| \)-subseparable. Yet, in Sect. 4 we show that the value of parameter \( p \) in many natural problems is significantly lower.

We observe that a linear combination of p-superseparable functions is p-superseparable. The same comment applies to p-subseparability. As we will see in Sect. 4, this observation is helpful in proving that certain set functions are p-separable.

In this paper we investigate the problem of selecting \( K \) elements from \( X \) that, altogether, maximize the value of the set function \( v \).

Definition 2 (BestKSubset). For a set of elements \( X \), a polynomially computable set function \( v : 2^X \rightarrow \mathbb{R} \), and an integer \( K \), the solution to the BestKSubset problem is such a set \( S \subseteq X \) that \( |S| \leq K \) and that \( v(S) \) is maximal.

We are specifically interested in finding approximation algorithms for the BestKSubset problem. We focus on approximating two metrics: (i) the value \( v(S) \) in the maximization variant of the problem, and (ii) the value \( (v(X) - v(S)) \) in its minimization variant.

Definition 3 (Approximation algorithms). Let \( S^* \) denote an optimal solution for BestKSubset:

1. Fix \( \alpha \), \( 0 < \alpha < 1 \). \( \mathcal{A} \) is an \( \alpha \)-approximation algorithm for the maximization variant of BestKSubset, if for each instant \( I \) of BestKSubset it returns a set \( S \) such that \( v(S) \geq \alpha v(S^*) \).

2. Fix \( \alpha \), \( \alpha > 1 \). \( \mathcal{A} \) is an \( \alpha \)-approximation algorithm for the minimization variant of BestKSubset, if for each instant \( I \) of BestKSubset it returns a set \( S \) such that \( (v(X) - v(S)) \leq \alpha (v(X) - v(S^*)) \).

3. Fix \( \alpha \), \( \alpha > 1 \). \( \mathcal{A} \) is an \( \alpha \)-approximation algorithm for the minimization-or-maximization variant of BestKSubset, if for each instant \( I \) of BestKSubset it returns a set \( S \) such that \( v(S) \geq \frac{1}{\alpha} v(S^*) \) or \( (v(X) - v(S)) \leq \alpha (v(X) - v(S^*)) \).
The definition of an approximation algorithm for minimization-or-maximization variant of \textsc{BestKSubset} requires some additional comment: this definition guarantees that the algorithm finds a good solution provided a high quality solution exists. In other words, if there exists an optimal solution $S^*$ such that the value $(v(X) - v(S^*))$ is low compared to $v(S^*)$, then the good approximation solution for the minimization variant of the problem is also a good solution for its maximization variant. For some parameters we present good approximation algorithms for the minimization-or-maximization variant of \textsc{BestKSubset}, even though we do not have as good algorithms neither for the minimization nor maximization variants of the problem.

We are specifically interested in FPT approximation schemes. A collection of algorithms $\mathcal{A}$ is an FPT approximation scheme for a parameter $P$, if for each constant $\alpha$ there exists an $\alpha$-approximation algorithm in $\mathcal{A}$ that runs in an FPT time for the parameter $P$.

3 Algorithms for Maximizing p-Separable Submodular Functions

In this section we present our approximation algorithms for the two variants of the problem, formally stated in Definition 3, of the \textsc{BestKSubset} problem. Our methods are inspired by the algorithms of Skowron and Faliszewski \cite{27} for the \textsc{MaxCover} problem. We extend these algorithms to be applicable to the problem of maximizing more general submodular functions.

We start with presenting an FPT approximation scheme for \textsc{BestKSubset} for submodular $p$-superseparable set functions. The algorithm, formally defined as Algorithm 1, gets as an input an instance of the problem and the required approximation ratio, $\beta$. It proceeds in two steps: first, it restricts the universe set by selecting a certain number of elements from $X$ with the highest values of the set function $v$. Second, it takes the set $A$ of elements that were selected in the first step, computes the value of the set function for all $K$-element subsets of $A$, and returns a subset with the highest value.

Algorithm 1 is an FPT approximation scheme for the maximization variant of the problem for the parameter $(K, p)$. Before we prove this fact, however, we note that under standard complexity theoretic assumptions, there exists no FPT

\begin{algorithm}
\caption{An algorithm for the \textsc{BestKSubset} problem for non-negative, non-decreasing, submodular, and $p$-superseparable set functions.}
\begin{algorithmic}
\STATE Parameters:
\STATE $X$ — the set of elements.
\STATE $v$ — the submodular function $v : 2^X \rightarrow \mathbb{R}$ that is $p$-superseparable.
\STATE $\beta$ — the required approximation ratio of the algorithm.
\STATE $A \leftarrow \lceil \frac{pK}{(1-\beta)} + K \rceil$ elements $x$ from $X$ with highest values $v(\{x\})$;
\STATE return $K$-element subset of $A$ with the highest value of $v$;
\end{algorithmic}
\end{algorithm}
exact algorithm for the problem. There even exists no FPT exact algorithm for the parameter $K$ if $p$ is a constant. This follows from our observation in Sect. 4.1, where we show that the MaxCover problem with frequencies bounded by $p$ can be expressed as maximization of a non-negative, non-decreasing, submodular, $p$-superseparable set function, and from the fact that the MaxCover problem with frequencies bounded by a constant, for the parameter $K$ belongs to the complexity class $W[1]$ [27], and it is unlikely that $W[1] \subseteq$ FPT.

**Theorem 1.** For each non-negative, non-decreasing, submodular, and $p$-superseparable set function $v : 2^X \rightarrow \mathbb{R}$ and for each $0 \leq \beta < 1$, Algorithm 1 outputs a $\beta$-approximate solution for the maximization variant of BestKSubset, in time $\text{poly}(n, m) \cdot \left(\frac{pK}{m} + K\right)$.

**Proof.** Consider an input instance $I$ of the BestKSubset problem. Let $S$ and $S^*$ be, respectively, the solution returned by Algorithm 1 and some optimal solution. We set $\text{OPT} = v(S^*)$ as the value of an optimal solution.

We will show that $v(S) \geq \beta \text{OPT}$. Naturally, the value $v(S)$ might be lower than $v(S^*)$. This might happen because $A$, the set of the elements considered by the algorithm in its second step, might not contain some elements from $S^*$. We will show that $\ell = |S^* \setminus A|$ elements from $S^* \setminus A$ might be replaced by some elements from $A$ which are not present in $S^*$, in a way that decreases the value of $S^*$ by at most a small fraction. After such replacement, we will end up with the set containing the elements from $A$ only. From this we will infer that the value of the best solution in $A$ is lower than the value of an optimal solution by at most a small factor.

Let us order the elements from $S^* \setminus A$ in some arbitrary way, and let us use the notation $S^* \setminus A = \{x_1, \ldots, x_\ell\}$. We will replace the elements $\{x_1, \ldots, x_\ell\}$ with the elements $\{x'_1, \ldots, x'_\ell\}$ (we will define these elements later), one by one, in $\ell$ consecutive steps. Thus, in the $i$-th step we will replace $x_i$ with $x'_i$ in the set $(S^* \setminus \{x_1, \ldots, x_{i-1}\}) \cup \{x'_1, \ldots, x'_i\}$. The elements $x'_1, \ldots, x'_i$ are defined by induction, in the following way. Assume that we have already found elements $x'_1, \ldots, x'_{i-1}$ (for $i = 1$ it means we have not yet found any element, i.e., that we are looking for the first element in the sequence). We define $x'_i$ to be an element from $A \setminus (S^* \cup \{x'_1, \ldots, x'_{i-1}\})$ that maximizes the value $v((S^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_i\})$.

It may happen that after replacing $x_i$ with $x'_i$, the value of the function $v$ for the new set decreases. Let $\Delta_i$ denote the value of such decrease (or increase if the algorithm were lucky—in such case $\Delta_i$ would be negative):

$$\Delta_i = v\left((C^* \setminus \{x_1, \ldots, x_{i-1}\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) - v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_i\}\right).$$

By the construction of the set $A$ and the fact that $x_i \not\in A$, for every $y \in A \setminus (S^* \cup \{x'_1, \ldots, x'_{i-1}\})$ we have that $v(\{x_i\}) \leq v(\{y\})$. By the way we choose the element $x'_i$, we know that for every $y \in A \setminus (S^* \cup \{x'_1, \ldots, x'_{i-1}\})$, we have:
\[ \Delta_i \leq v\left((C^* \setminus \{x_1, \ldots, x_{i-1}\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) \\
- v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}, y\}\right). \]

Using submodularity and after reformulation we get:

\[ \Delta_i \leq v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) + v(\{x_i\}) - v(\emptyset) \\
- v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}, y\}\right) \leq v\left((C^* \setminus \{x_1, \ldots, x'_i\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) + v(\{y\}) - v(\emptyset) \\
- v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}, y\}\right). \]

For any \( y \in X \) (in particular for \( y \notin A \setminus (S^* \cup \{x'_1, \ldots, x'_{i-1}\}) \)), by submodularity and monotonicity, we have that:

\[ 0 \leq v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) + v(\{y\}) - v(\emptyset) \\
- v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}, y\}\right). \]

Since the set function is non-negative, the inequalities above will still hold if we skip the fragment \( v(\emptyset) \). Consequently, since the set function is \( p \)-superseparable, we get:

\[ (|A| - K) \Delta_i \leq \sum_{y \in X} \left(v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) + v(\{y\}) \\
- v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}, y\}\right)\right) \leq p \cdot v\left((C^* \setminus \{x_1, \ldots, x_i\}) \cup \{x'_1, \ldots, x'_{i-1}\}\right) \leq p \cdot \text{OPT}. \]

Which leads to:

\[ \Delta_i \leq \frac{p \cdot \text{OPT}}{|A| - K} = \frac{\text{OPT}p(1 - \beta)}{pK} = \frac{\text{OPT}(1 - \beta)}{K}. \]

Since \( \ell \leq K \), we conclude that:

\[ \sum_{i=1}^{\ell} \Delta_i \leq (1 - \beta) \text{OPT}. \]

That is, after replacing the elements from \( S^* \) that do not appear in \( A \) with sets from \( A \), the optimal value is decreased by at most \((1 - \beta) \text{OPT}\). This means that there are \( K \) elements in \( A \) for which the function \( v \) achieves the value equal to at least \( \beta \text{OPT} \). Since the algorithm tries all size-\( K \) subsets of \( A \), it finds a solution with such a value. \( \square \)
Algorithm 2. An algorithm for the minimization variant of the BestKSubset problem with a non-negative, non-decreasing, submodular, and $p$-subseparable set function.

Parameters:
$X$ — the set of elements.
$v$ — the submodular function $v: 2^X \to \mathbb{R}$ that is $p$-subseparable.
$\beta$ — the required approximation ratio of the algorithm
$\epsilon$ — the allowed probability of achieving worse than $\beta$ approximation ratio

SingleRun():

$S \leftarrow \emptyset$;
for $i \leftarrow 0$ to $K$ do

$x_r \leftarrow$ randomly select an element from $X \setminus S$

with probability of selecting $x$ proportional to $v(S \cup \{x\}) - v(S)$;

$S \leftarrow S \cup \{x_r\}$;

return $S$;

Main(): run SingleRun() for $\lceil -\ln \epsilon / (\beta - 1)p\beta K \rceil$ times; return the best solution;

Next, we consider the minimization variant of BestKSubset for the case of $p$-subseparable submodular set functions. In Algorithm 2 we present a randomized algorithm for the problem: the algorithm performs several independent runs. Each run, in Algorithm 2 described by the SingleRun procedure, builds the solution by selecting random elements in $K$ consecutive steps. In each step, an element $x$ is selected with the probability proportional to the marginal increase of the value of the set function caused by adding $x$ to the partial solution. Theorem 2 below shows that if we repeat SingleRun a sufficient number of times, we are very likely to find a solution with the required approximation ratio.

Theorem 2. For each non-negative, non-decreasing, submodular, $p$-subseparable set function $v: 2^X \to \mathbb{R}$ and for each $0 \leq \beta < 1$, Algorithm 2 outputs a $\beta$-approximate solution for the minimization variant of BestKSubset, with probability $(1 - \epsilon)$. The time complexity of the algorithm is $\text{poly}(n, m) \cdot \lceil -\ln \epsilon / (\beta - 1)p\beta K \rceil$.

Proof. Let $I$ be an instance of the BestKSubset problem with $v: 2^X \to \mathbb{R}$ being a non-negative, submodular, $p$-subseparable function. Let $\beta, \beta > 1$, and $\epsilon, 0 < \epsilon < 1$ be the parameters of Algorithm 2. Let $S^*$ be some optimal solution for $I$.

Let us consider a single call to SingleRun from the “for” loop within the function Main. Let $p_s$ denote the probability that such a single invocation of the function SingleRun returns a $\beta$-approximate solution. We will prove the lower-bound of $(\beta - 1)p\beta K$ for the value of $p_s$. Let $Ev$ denote the event that during such an invocation, at the beginning of each iteration of the “for” loop within the function SingleRun, it holds that:

$$v(X) - v(S) > \beta \left( v(X) - v(S^*) \right).$$  (3)
Note that if the complementary event, denoted $\overline{E_v}$, occurs, then $\text{SingleRun}$ definitely returns a $\beta$-approximate solution. The condition in Inequality 3 can be reformulated as follows:

$$\frac{v(S^*) - v(S)}{v(X) - v(S)} > \frac{\beta - 1}{\beta}. \tag{4}$$

Now, let us consider a single iteration of the “for” loop within the function $\text{SingleRun}$. Let $S$ be the value of the partial solution at the beginning of this iteration and let $p_{\text{hit}}$ denote the probability that in this iteration the element from $S^*$ is added to the partial solution (thus, using notation from Algorithm 2, $p_{\text{hit}}$ is the probability that $x_r \in S^*$). Let us assess the conditional probability $p_{\text{hit}|Ev}$:

$$p_{\text{hit}|Ev} = \frac{\sum_{x \in S^*} (v(S \cup \{x\}) - v(S))}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} = \frac{v(S \cup S^*) - v(S)}{v(S^*) - v(S)}$$

$$\geq \frac{v(S^*) - v(S)}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} \text{ non-decreasing}$$

$$\geq \frac{v(S^*) - v(S)}{p(v(X) - v(S))} \quad p\text{-subseparability}$$

$$\geq \frac{\beta - 1}{p\beta}. \quad \text{Eq. 4}$$

Let $p_{\text{opt}}$ denote the probability that the function $\text{SingleRun}$ returns $S^*$, an optimal solution. We have that:

$$p_{\text{opt}|Ev} \geq \left( p_{\text{hit}|Ev} \right)^K \geq \left( \frac{\beta - 1}{p\beta} \right)^K.$$ 

Altogether, combining all the above findings, we know that the probability that $\text{SingleRun}$ returns a $\beta$-approximate solution is at least:

$$p_s \geq P(\overline{E_v}) + P(E_v)p_{\text{opt}|Ev} \geq p_{\text{opt}|Ev} \geq \left( \frac{\beta - 1}{p\beta} \right)^K. \tag{5}$$

The estimation in Inequality 5 can be obtained by observing that either the event $E_v$ or $\overline{E_v}$ must happen. If $E_v$ happens, then $\text{SingleRun}$ definitely returns a $\beta$-approximate solution; if $E_v$ happens, then we can lower-bound the probability of finding a $\beta$-approximate solution by the probability of finding an optimal one.
To conclude, we use the standard argument that if we make \( x = \left\lceil -\frac{\ln \epsilon}{p_s} \right\rceil \) independent calls to \textit{SingleRun}, then the best output from these calls is a \( \beta \)-approximate solution with probability at least equal to:

\[
1 - (1 - p_s)^x \geq 1 - e^{\ln \epsilon} = 1 - \epsilon.
\]

This completes the proof. \( \square \)

Interestingly, we can slightly modify the proof of Theorem 2 so that it would apply with the more general parameter \( \sum_{x \in X} \frac{v(\{x\})}{v(X)} \). On the other hand, for this parameter we give weaker approximation guarantees, by approximating the minimization-or-maximization instead of the minimization variant of the problem.

**Theorem 3.** For each non-negative, non-decreasing and submodular set function \( v : 2^X \rightarrow \mathbb{R} \) there exists an FPT approximation scheme for the minimization-or-maximization variant of \textsc{BestKSubset} problem with the parameter \( (K, \frac{\sum_{x \in X} v(\{x\})}{v(X)}) \).

**Proof.** Let us fix \( \beta, \beta > 1 \), the required approximation ratio. Let \( p = \frac{\beta}{\beta - 1} \cdot \frac{\sum_{x \in X} v(\{x\})}{v(X)} \). We will show that Algorithm 2 with such value of the parameter \( p \) (this parameter is used to determine the number of iterations of the algorithm) is a \( \beta \)-approximation algorithm for the minimization-or-maximization variant of the problem. We repeat the reasoning from the proof of Theorem 2, with the following small modification. In the proof of Theorem 2 we defined \( Ev \) to denote the event that during a single invocation of the \textit{SingleRun} function from Algorithm 2, at the beginning of each iteration of the “for” loop, it holds that:

\[
v(X) - v(S) > \beta \left( v(X) - v(S^*) \right).
\]

In this proof we modify this definition saying that \( Ev \) denotes the event when at the beginning of each iteration of the “for” loop within the function \textit{SingleRun}, the following two conditions hold:

\[
v(X) - v(S) > \beta \left( v(X) - v(S^*) \right),
\]

\[
v(S) < \frac{1}{\beta} v(S^*).
\]

Naturally, if the complementary event occurs, then \textit{SingleRun} definitely returns a \( \beta \)-approximate solution for the minimization-or-maximization variant of the problem. In the proof of Theorem 2, we used at-most-\( p \)-subseparability in the part that assumes that the event \( Ev \) happened, to show that:

\[
\sum_{x \in X} \left( v(S \cup \{x\}) - v(S) \right) \leq p \left( v(X) - v(S) \right)
\]

Here, we show that Inequality 6 also holds if we assume that the event \( Ev \) (using our redefinition of \( Ev \)) happened:
\[
\sum_{x \in X} \left( v(S \cup \{x\}) - v(S) \right) \leq \sum_{x \in X} \left( v(\{x\}) - v(\emptyset) \right) \leq \sum_{x \in X} v(\{x\})
\] 
\[= p \cdot \beta - 1 \cdot \frac{1}{\beta} \cdot v(X) = p \cdot v(X) - p \cdot \frac{v(X)}{\beta} \]
\[\leq p \cdot v(X) - p \cdot v(S).\]

With these modifications the proof of Theorem 2 can be used in this case. \[\square\]

Algorithm 2 can be applied to yet another variant of the problem. Let BESTSUBSET be defined similarly to BESTKSUBSET, with the following difference. In BESTSUBSET we are not putting any constraints on the size of the solution, but we rather look for the smallest possible set \( S \) such that \( v(S) = v(X) \). Interestingly, Algorithm 2 can be used to find exact solutions to BESTSUBSET for non-negative, non-decreasing, submodular, \( p \)-subseparable set functions, and it will run in FPT time for the parameter \((K, p)\).

**Theorem 4.** For each non-negative, non-decreasing, submodular, \( p \)-subseparable set function \( v : 2^X \to \mathbb{R} \), the algorithm that runs Algorithm 2 for consecutive values of the parameter \( K \) until it finds a solution \( S \), such that \( v(S) = v(X) \), is a randomized FPT exact algorithm for the BESTSUBSET problem for the parameter \((K, p)\).

**Proof.** The proof is provided in the full version of the paper [26].

### 4 Applications of the Algorithms

In this section we show that the assumption about \( p \)-separability of submodular set functions is plausible. We provide several examples of known computational problems that can be expressed as maximization of \( p \)-separable, submodular functions.

#### 4.1 The MaxWeightCover Problem

In this subsection we show that our algorithms are applicable to MAXWEIGHTCOVER, a generalized variant of the MAXCOVER problem.

In the MAXWEIGHTCOVER problem, we are given a universe set \( N = \{e_1, e_2, \ldots, e_n\} \) of \( n \) elements and a collection \( X = \{S_1, \ldots, S_m\} \) of \( m \) subsets of \( N \). Each element \( e_i \) has its weight \( w_i \). The goal is to find a subcollection \( C \) of \( X \) of size at most \( K \) that maximizes the total weight of covered elements: \[
\sum_{i : i \in S \text{ for some } S \in C} w_i.
\]

A frequency of an element \( e_i \) is the number of sets that contain \( e_i \). Frequency of elements is a natural parameter considered in the context of approximability of covering problems [32]. To the best of our knowledge, for polynomial-time algorithms, there exists no better guarantee for the MAXCOVER problem with bounded frequencies of elements than \((1 - 1/e)\). This is specifically interesting, since such an approximation algorithm exists for the very similar problem SETCOVER [32].
Lemma 1. The MaxWeightCover problem with the frequency of elements upper-bounded by $p$ can be expressed as the maximization of a nonnegative, nondecreasing submodular function which is (i) $p$-superseparable, and (ii) $p$-subseparable.

Proof. For each set $C \subseteq X$ we define $v(C)$ as the total weight of elements covered by the sets from $C$. Such defined $v$ is nonnegative and submodular.

We observe that the weighted sum of $p$-superseparable set functions is also $p$-superseparable, and that the same argument applies to $p$-subseparability. Thus, it is sufficient to consider a function $u_i$ which returns 1 for collections of sets that cover $e_i$, and 0 for the remaining ones. Observe that if the frequency of the elements is bounded by $p$, then $\sum_{S \in X} u_i(\{S\})$, the number of sets that cover $e_i$, is also bounded by $p$.

Let us fix a collection of sets $C \subseteq X$ and let us consider two cases. If $e_i$ is covered by $C$, then $\sum_{S \in X} (u_i(C \cup \{S\}) - u_i(C))$ is equal to 0. But, in such case $pu_i(C) = p$ and the condition for $p$-superseparability holds. Naturally, $u_i(X) = 1$, thus the conditions for $p$-subseparability also holds.

If $e_i$ is not covered by $C$, then $\sum_{S \in X} (u_i(C \cup \{S\}) - u_i(C))$ is equal to the number of sets that cover $e_i$, thus to $\sum_{S \in X} u_i(\{S\})$. This means that the condition for $p$-superseparability holds. If the frequency of the elements is upper-bounded by $p$, then $\sum_{S \in X} (u_i(C \cup \{S\}) - u_i(C))$ is upper bounded by $p$, and since $u_i(C) = 0$, the condition for $p$-subseparability holds. This proves the thesis. $\square$

Corollary 1. There exists an FPT approximation scheme for the maximization and minimization variant of the MaxWeightCover for the parameter $(K,p)$, where $p$ is the upper-bound on the frequency of the elements.

Corollary 1 extends the recent results for the MaxCover problem [27]. Interestingly, Theorem 3 says that there exists a randomized FPT approximation scheme for the minimization-or-maximization variant of the MaxWeightCover problem, for the parameter $(K,p_{av})$, where $p_{av}$ is an average frequency of an element.

4.2 Other Applications

Due to space restrictions in this section we describe the other two applications of our results very briefly. For the thoughtful analysis of these two cases we refer the reader to the full version of the paper [26].

Matching and assignment problems. In the Weighted-B-K-Matching problem we are given a set of vertices $X \cup Y$, a set of edges $E$ (there are no edges neither between the vertices from $X$ nor between the vertices from $Y$), a weight function $w : E \to \mathbb{R}$, and a capacity function $c : X \to \mathbb{Z}$. The goal is to find a subset of edges with the maximal total weight, such that each vertex $x \in X$ belongs to at most $c(x)$ of the selected edges, each vertex $y \in Y$
belongs to at most one of the selected edges, and altogether there are at most $K$ vertices from $X$ which belong to some of the selected edges.

Our results can be used to prove that there exists an FPT approximation scheme for the maximization variant of the Weighted-B-$K$-Matching for the parameter $(K, p)$, where $p$ is a bound on the degree of vertices from $Y$.

**Item selection in multi-agent systems.** Our results can be also applied to the remarkably general model describing the problem of selecting a set of collective items for agents [28]. Let $N = \{1, 2, \ldots, n\}$ be the set of agents and let $C = \{a_1, a_2, \ldots, a_m\}$ be the set of items. Each agent $i \in N$ is endowed with a utility function $u_i : C \rightarrow \mathbb{R}$ that measures how much $i$ desires each of the items. Our goal is to select $K$ items, called winners, that in some sense would make the agents most satisfied. An OWA vector $\alpha$ is a vector of $K$ elements, $\alpha = (\alpha_1, \ldots, \alpha_K)$. Given an OWA vector $\alpha$, for each agent $i$ and for each set of $K$ items $S$, we define $u_i(S)$, the satisfaction of $i$ from $S$, in the following way. Let $u_1, u_2, \ldots, u_K$ be the utilities from $\{u_i(x) : x \in S\}$, sorted in the descending order; then $u_i(S) = \sum_{j=1}^{K} \alpha_j u_j$. The satisfaction of all agents from $S$ is defined as the sum of satisfactions of all the individuals from $S$.

This model captures various natural problems, from winner determination in multiwinner election systems, through recommendation systems, to location problems. For instance the problem of selecting $K$ items under the OWA vector $\alpha = (1, 0, \ldots, 0)$ boils down to the problem of winner determination under Chamberlin and Courant rule [3], or to the facility location problem. The problem for $\alpha = (1, 1/2, \ldots, 1/K)$ is equivalent to winner determination in the Proportional Approval Voting (PAV) system. For more examples of applications of this general model we refer the reader to the original work of Skowron et al. [28].

We say that the agents have $k$-approval utilities if each agent assigns utility equal to 1 to exactly $k$ items, and utility equal to 0 to the remaining ones. Such $k$-approval utilities are very popular in the context of social choice, in particular in case of multi-winner election rules.

Our results can be used to prove that there exists an FPT approximation scheme for the maximization and minimization variants of the problem of selecting $K$ items with $k$-approval utilities for the parameter $(K, k)$.

5 Conclusions

We have considered FPT approximation schemes for the problem of maximizing submodular set functions. There are many natural ways in which this research can be extended. We believe that one of the promising approaches is to consider the problem with additional constraints, such as knapsack constraints or matroid constraints.

**Acknowledgements.** The author thanks Piotr Faliszewski for his helpful comments. This research has been supported by Europe Research Grant ERC-StG 639945.
References

1. Bach, F.: Structured sparsity-inducing norms through submodular functions. In: Proceedings of Advances in Neural Information Processing Systems 23 (NIPS 2010), pp. 118–126 (2010)
2. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint (extended abstract). In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 182–196. Springer, Heidelberg (2007). doi:10.1007/978-3-540-72792-7_15
3. Chamberlin, B., Courant, P.: Representative deliberations and representative decisions: proportional representation and the borda rule. Am. Polit. Sci. Rev. 77(3), 718–733 (1983)
4. Chvatal, V.: A greedy heuristic for the set-covering problem. Math. Oper. Res. 4(3), 233–235 (1979)
5. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer, Heidelberg (2015)
6. Das, A., Kempe, D.: Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In: Getoor, L., Scheffer, T. (eds.) Proceedings of the 28th International Conference on Machine Learning (ICML 2011), New York, NY, USA, pp. 1057–1064 (2011)
7. Downey, R., Fellows, M.: Parameterized Complexity. Springer, New York (1999)
8. Feige, U.: A threshold of \( \ln n \) for approximating set cover. J. ACM 45(4), 634–652 (1998)
9. Feige, U., Mirrokni, V.S., Vondrák, J.: Maximizing non-monotone submodular functions. SIAM J. Comput. 40(4), 1133–1153 (2011)
10. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2006)
11. Iwata, S., Fleischer, L., Fujishige, S.: A combinatorial strongly polynomial algorithm for minimizing submodular functions. J. ACM 48(4), 761–777 (2001)
12. Jegelka, S., Bilmes, J.A.: Submodularity beyond submodular energies: coupling edges in graph cuts. In: Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011), pp. 1897–1904 (2011)
13. Kim, G., Xing, E.P., Fei-Fei, L., Kanade, T.: Distributed cosegmentation via submodular optimization on anisotropic diffusion. In: Proceedings of IEEE International Conference on Computer Vision (ICCV 2011), pp. 169–176 (2011)
14. Krause, A., Golovin, D.: Submodular function maximization. Technical report (2012)
15. Krause, A., Guestrin, C.: Near-optimal value of information in graphical models. In: Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI 2005), July 2005
16. Krause, A., Singh, A., Guestrin, C.: Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. J. Mach. Learn. Res. 9, 235–284 (2008)
17. Kumar, R., Moseley, B., Vassilvitskii, S., Vattani, A.: Fast greedy algorithms in MapReduce and streaming. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2013), pp. 1–10 (2013)
18. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Non-monotone submodular maximization under matroid and knapsack constraints. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing (STOC 2009), pp. 323–332 (2009)
19. Lin, H., Bilmes, J.: Multi-document summarization via budgeted maximization of submodular functions. In: Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HTL 2010), pp. 912–920. Association for Computational Linguistics, Stroudsburg (2010)

20. Lin, H., Bilmes, J.A.: How to select a good training-data subset for transcription: submodular active selection for sequences. In: Proceedings of 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009), pp. 2859–2862 (2009)

21. Lu, T., Boutilier, C.: Budgeted social choice: from consensus to personalized decision making. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011), pp. 280–286 (2011)

22. Narasimhan, M., Nebojsa, J., Jeff, B.A.: Q-clustering. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Proceedings of Advances in Neural Information Processing Systems 18 (NIPS 2010), pp. 979–986. MIT Press (2006)

23. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of approximations for maximizing submodular set functions. Math. Program. 14(1), 265–294 (1978)

24. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, Oxford (2006)

25. Jegelka, S., Bach, F., Sra, S.: Reflection methods for user-friendly submodular optimization. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (eds.) Proceedings of Advances in Neural Information Processing Systems 26 (NIPS 2013), pp. 1313–1321. Curran Associates Inc. (2013)

26. Skowron, P.: FPT approximation schemes for maximizing submodular functions. Technical report arXiv:1510.00215 [cs.DS], January 2015

27. Skowron, P., Faliszewski, P.: Fully proportional representation with approval ballots: approximating the maxcover problem with bounded frequencies in FPT time. In: Proceedings of the 29th Conference on Artificial Intelligence (AAAI 2015), pp. 2124–2130 (2015)

28. Skowron, P., Faliszewski, P., Lang, J.: Finding a collective set of items: from proportional multirepresentation to group recommendation. Artif. Intell. 241, 191–216 (2016)

29. Skowron, P., Faliszewski, P., Slinko, A.: Achieving fully proportional representation: approximability result. Artif. Intell. 222, 67–103 (2015)

30. Streeter, M.J., Golovin, D.: An online algorithm for maximizing submodular functions. In: Proceedings of Advances in Neural Information Processing Systems 21 (NIPS 2008), pp. 1577–1584 (2008)

31. Sviridenko, M.: A note on maximizing a submodular set function subject to a knapsack constraint. Oper. Res. Lett. 32(1), 41–43 (2004)

32. Vazirani, V.V.: Approximation Algorithms. Springer, New York (2001)

33. Vondrák, J., Chekuri, C., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. In: Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing (STOC 2011), pp. 783–792 (2011)

34. Yue, Y., Guethrin, C.: Linear submodular bandits and their application to diversified retrieval. In: Proceedings of Advances in Neural Information Processing Systems 24 (NIPS 2011), pp. 2483–2491 (2011)