Greedy Algorithms with Imprecise Oracles for Submodular Knapsack Problems

Sabine Münch, Stephen Raach

Trier University, 54286 Trier, Germany

{muench/raach}@uni-trier.de

We consider the problem of maximizing a monotone increasing, normalized, and submodular function defined on a set of weighted items under a knapsack constraint. A well-known greedy algorithm, analyzed by Wolsey (1982), achieves an approximation factor of 0.357 for this problem. This greedy algorithm starts with an empty solution set and iteratively generates a feasible solution by packing items with maximum relative marginal gain to the current solution, stopping when adding the next item would violate the knapsack constraint. Then it compares the objective value of the current solution to that of the element that was last attempted to be packed and returns the one with the greater objective value. Hence, it relies on the existence of an exact polynomial-time incremental oracle that identifies, in each iteration, an item with maximum relative marginal gain to the current solution.

We generalize the known approximation result for this greedy algorithm to the presence of only an approximate incremental oracle that identifies in each iteration of the greedy algorithm an item whose relative marginal gain approximates the maximum relative marginal gain by at least $\frac{1}{\alpha}$, with $\alpha \geq 1$ fixed. Furthermore, we present an approximation result for a variant of the greedy algorithm in which an approximate incremental oracle is used in the first iteration of the greedy algorithm, and an exact incremental oracle is used in all subsequent iterations.

Keywords : *Submodular maximization, knapsack, greedy algorithm, approximate incremental oracle*

1. Introduction.

Submodular functions play a central role in many classical combinatorial optimization problems including graph cuts (see, e.g., Goemans and Williamson 1995), facility location (see, e.g., Ageev and Sviridenko 1999; Cornuejols et al. 1977), generalized assignment (see, e.g., Feige and Vondrák 2006; Fleischer et al. 2006), and set cover (see, e.g., Krause et al. 2008; Feige 1998).

Moreover, the problem of maximizing a submodular function under various constraints arises naturally in many application domains, as submodular functions inherently capture the property of diminishing returns. These applications include classical machine learning tasks such as extractive document and image summarization (see, e.g., Lin and Bilmes 2011; Tschiatschek et al. 2014) and data subset selection (see, e.g., Wei et al. 2015), as well as broader AI-related optimization problems like sensor placement (see, e.g., Krause and Guestrin 2007; Krause et al. 2008) and influence maximization in social networks (see, e.g., Kempe et al. 2015).

Recall that a function $f\colon 2^I \to \mathbb{R}_{\geq 0}$ defined over a finite set of elements I is called normalized if $f(\emptyset) = 0$, monotone increasing if $f(X) \leq f(Y)$ for any $X \subseteq Y \subseteq I$, and submodular if $f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$ for $X, Y \subseteq I$.

Definition 1. *In the following, let I be a finite set of cardinality $n \in \mathbb{N}$, and let $f\colon 2^I \to \mathbb{R}_{\geq 0}$ be a **normalized**, **monotone increasing** and **submodular** function with $f(\{i\}) > 0$ for $i \in I$. To simplify the notation, we define $f(i) \coloneqq f(\{i\})$ for every $i \in I$. Further, we call any $i \in I$ an **item** and assume that every item $i \in I$ is associated with a **weight** $w_i \in \mathbb{R}_{>0}$. We write $w(X) \coloneqq \sum_{i \in X} w(i)$ for the weight of a set $X \subseteq I$.*

We consider the classic problem of *maximizing a submodular function subject to a knapsack constraint* (SMKC), thus,

$$\max\{f(X)\colon w(X) \leq B,\ X \subseteq I\}, \tag{1}$$

where $B \in \mathbb{R}_{\geq 0}$ is called **knapsack capacity**.

An instance of this problem is given by a tuple (I, f, w, B), where $w = (w_i)_{i \in I}$ is a vector of weights with $w_i > 0$ for all items $i \in I$, and $B \in \mathbb{R}_{\geq 0}$ is a knapsack capacity.

Definition 2. *For any instance (I, f, w, B), we call a set $X \subseteq I$ an **optimal solution** if $w(X) \leq B$ and $f(X)$ attains the value of (1). We refer to the objective value of an optimal solution to an instance (I, f, w, B) as $\mathrm{Opt}(I, f, w, B)$.*

It is well known that SMKC is an NP-hard problem. However, there are several polynomial-time algorithms that, for any instance of SMKC, compute a feasible solution whose objective value approximates the value of an optimal solution by a constant factor.

Definition 3. *Let $\mathcal{A}$ be an algorithm that returns a feasible solution $S \subseteq I$ for any* SMKC*-instance (I, f, w, B). We define the approximation factor of $\mathcal{A}$ as*

$$\min_{\substack{\text{SMKC-}\textit{instance } (I,f,w,B) \\ \textit{with } \mathrm{Opt}(I,f,w,B) \neq 0}} \frac{f(S)}{\mathrm{Opt}(I, f, w, B)}.$$

Notice that the approximation factor of an algorithm $\mathcal{A}$ that returns a feasible solution to SMKC is well-defined, since division by zero is excluded.

Among polynomial-time algorithms with a constant approximation factor for SMKC, a simple greedy algorithm analyzed by Wolsey (1982) stands out.

Definition 4. *Let (I, f, w, B) be an* SMKC*-instance. For any item $i \in I$ and $X \subseteq I$, we define the* ***marginal gain*** *of i to X by $f(i|X) \coloneqq f(\{i\} \cup X) - f(X)$ and the* ***relative marginal gain*** *of i to X by $\frac{f(i|X)}{w_i}$.*

Given an SMKC-instance, the greedy algorithm studied by Wolsey (1982) starts with an empty solution set and iteratively constructs a feasible solution by selecting, in each iteration, the item with maximum relative marginal gain for addition to the current solution. Items are added until the next selected item does not fit into the knapsack, i.e., its addition would cause the total weight of the packed items to exceed the knapsack capacity. Then the greedy algorithm compares the objective value of the current solution with that of the item that was last attempted to be added and returns the one with the greater objective value.

Wolsey (1982) showed that this greedy algorithm achieves an approximation factor of at least $1 - e^{-\beta} = 0.357\ldots$, where β is the unique solution to $e^x = 2 - x$, $x \in [0, 1]$.

The performance of this greedy algorithm relies on selecting, in each iteration, an item with maximum relative marginal gain to the current solution. This selection is carried out using an oracle.

Definition 5. *Let (I, f, w, B) be an* SMKC*-instance. Given a subset $R \subseteq I$, an* ***exact incremental oracle*** *selects an item $i^* \in I \setminus R$ with $\frac{f(i^*|R)}{w_{i^*}} \geq \frac{f(i|R)}{w_i}$, for all $i \in I \setminus R$.*

However, in many situations, identifying an item with maximum relative marginal gain to a given set is computationally challenging. In such cases, it is a natural idea to relax the exactness requirement and instead use an approximate incremental oracle, which selects an item whose relative marginal gain is within a factor of $\frac{1}{\alpha}$ of the maximum, for some fixed $\alpha \geq 1$.

Definition 6. *Let (I, f, w, B) be an* SMKC*-instance and $\alpha \geq 1$. Given a subset $R \subseteq I$, an α****-approximate incremental oracle*** *selects an item $i^* \in I \setminus R$ with*

$$\frac{f(i^*|R)}{w_{i^*}} \geq \frac{1}{\alpha} \frac{f(i|R)}{w_i}, \quad \text{for all } i \in I \setminus R.$$

We refer to α as the ***approximation parameter*** *of an α-approximate incremental oracle.*

In the present paper, we focus on modified versions of the greedy algorithm, where the exact incremental oracle is replaced with an α-approximate incremental oracle.

1.1. Results.

We analyze the performance of two variants of the greedy algorithm studied by Wolsey (1982). In the first variant, the exact incremental oracle is replaced by an α-approximate incremental oracle in each iteration of the greedy algorithm. Generalizing the approximation result by Wolsey (1982) for the original greedy algorithm, we show that this modified greedy algorithm achieves an approximation factor of at least $1 - e^{-\frac{\gamma}{\alpha}}$, where γ is the unique solution to $e^{\frac{x}{\alpha}} = 1 + \frac{1-x}{\alpha}$, $x \in [0, 1]$.

In the second variant, the exact incremental oracle is replaced by an α-approximate incremental oracle in the first iteration of the greedy algorithm, and all other iterations use an exact incremental oracle. For this variant, we present an α-dependent approximation factor of at least $\min\left\{\frac{1-x}{2-x} \colon 2-x = \frac{\alpha}{\alpha-\lambda x} e^{(1-\lambda)x},\ \lambda \in (0,1]\right\}$.

1.2. Related work.

Research on SMKC was initiated by Nemhauser et al. (1978), who studied the problem of maximizing a submodular function subject to a cardinality constraint, an important special case of SMKC in which all items have unit weight. Notice that this special case of SMKC can also be interpreted as maximizing a normalized, monotone increasing, and submodular function subject to a uniform matroid. Nemhauser et al. (1978) have shown that a simple greedy algorithm, starting with an empty solution set and iteratively adding items with maximum marginal gain to the current solution until the cardinality constraint is reached, achieves an approximation factor of $1 - e^{-1} = 0.63\ldots$ for this special case of SMKC. Moreover, it was shown by Feige (1998) that, even in the special case with unit weights, no better approximation factor for SMKC can be achieved in polynomial time, unless $P = NP$.

As already mentioned, Wolsey (1982) studied a greedy algorithm for SMKC with arbitrary weights and demonstrated that this greedy algorithm achieves an approximation factor of at least $1 - e^{-\beta} = 0.357\ldots$, where β is the unique solution to $e^x = 2 - x$, $x \in [0,1]$.

Sviridenko (2004) showed that combining a partial enumeration scheme due to Sahni (1975) with this greedy algorithm achieves an approximation factor of $1-e^{-1} = 0.63\ldots$, thus matching the best possible approximation factor, unless $P = NP$.

Besides SMKC, the maximization of submodular functions over matroids and general independence systems has also been widely studied. Fisher et al. (1978) studied the maximization of a submodular function over a partition matroid and presented a variation of the greedy algorithm, called *locally greedy heuristic*, which achieves an approximation factor of $\frac{1}{2}$ for this problem. Further, they studied the maximization of a submodular function over an independence system that can be represented as the intersection of $M \in \mathbb{N}$ many matroids and presented a greedy algorithm for this problem that achieves an approximation factor of at least $\frac{1}{1+M}$.

All algorithms mentioned so far rely on the use of exact incremental oracles to approximately solve the respective problems. In contrast, Goundan and Schulz (2007) demonstrated that the greedy algorithm analyzed by Nemhauser et al. (1978), when using an α-approximate incremental oracle in place of an exact oracle, achieves an approximation factor of $1 - e^{-\frac{1}{\alpha}}$ for the submodular maximization problem subject to a uniform matroid, and that the locally greedy heuristic, modified in the same way, yields an approximation factor of $\frac{1}{\alpha+1}$ for the case of partition matroid constraints. In addition, they showed that, for maximizing a submodular function over an independence system representable as the intersection of M matroids, a greedy algorithm using an α-approximate incremental oracle achieves an approximation factor of $\frac{1}{\alpha M+1}$.

2. A greedy algorithm with an approximate incremental oracle for SMKC.

In this section, we focus on analyzing a modified version of the greedy algorithm studied by Wolsey (1982), as outlined below.

The modified version we analyze differs from the greedy algorithm studied by Wolsey (1982) by employing an approximate incremental oracle rather than an exact incremental oracle to select, in each iteration, the item to add to the current solution. Therefore, Algorithm 1 corresponds to the greedy algorithm analyzed by Wolsey (1982) when $\alpha = 1$.

Algorithm 1:

```
   Input: An SMKC-instance $(I, f, w, B)$ and an approximation parameter $\alpha \geq 1$.
   Output: Approximately optimal solution to SMKC.
1  $I \leftarrow I \setminus \{i \in I : w_i > B\}$
2  Set $R \leftarrow \emptyset$
3  while $I \setminus R \neq \emptyset$ do
4      Choose $i^* \in I \setminus R$ with $\frac{f(i^*|R)}{w_{i^*}} \geq \frac{1}{\alpha} \max\left\{\frac{f(i|R)}{w_i} \middle| i \in I \setminus R\right\}$ using an
         $\alpha$-approximate incremental oracle
5      if $w(R \cup \{i^*\}) \leq B$ then Set $R \leftarrow R \cup \{i^*\}$
6      else break
7  return  $\arg\max\{f(R), f(i^*)\}$
```

In the following, it is often useful to consider the items of the set R, constructed by Algorithm 1, in the order they were added to R. Thus, we assume that iteratively constructed sets are ordered according to the insertion order of all items.

Definition 7. *Let S be an arbitrary set of items constructed by iteratively adding items, starting with $S = \emptyset$, e.g., the set constructed in Line 4 of Algorithm 1. In the following, we assume that each such set S is ordered according to the insertion order of the items. For any $1 \leq j \leq |S|$, we denote by S_j the item added to S in the j-th iteration and by $S_{\leq j} := \{S_1, \ldots, S_j\}$ the set containing the first j items added to S. For $j = 0$, we define $S_{\leq 0} := \emptyset$.*

We fix some notation for the objective function value of the output of Algorithm 1 applied to an SMKC-instance (I, f, w, B) and an approximation parameter α.

Definition 8. *For any* SMKC*-instance (I, f, w, B) and approximation parameter $\alpha \geq 1$, we denote the objective function value of the return of Algorithm 1 as $\Phi(I, f, w, B, \alpha)$.*

The next two theorems (see Wolsey (1982)) provide well-known approximation guarantees for Algorithm 1 with $\alpha = 1$. We recall them here, as we aim to derive similar guarantees for Algorithm 1 with $\alpha > 1$.

Theorem 1. *(Wolsey 1982, Theorem 3) Algorithm 1 with $\alpha = 1$ has an approximation factor of*

$$1 - e^{-\beta} = 0.357\ldots,$$

where β is the unique solution to $e^x = 2 - x,\ x \in [0,1]$.

Theorem 2. *(Wolsey 1982, Theorem 2) Let (I, f, w, B) be an* SMKC*-instance and let R, with $w(R) = B$, be the set constructed in the while loop of Algorithm 1, applied to (I, f, w, B) and $\alpha = 1$. Then, for any knapsack capacity $D \geq B$, we have $\frac{f(R)}{\mathrm{Opt}(I,f,w,D)} \geq 1 - e^{-\frac{B}{D}}$.*

Note that Wolsey (1982) originally proved Theorems 1 and 2 for instances with integer weights. However, the results also extend to instances with arbitrary weights.

Before presenting approximation results for Algorithm 1, we recall an alternative characterization of monotone increasing submodular functions due to Nemhauser et al. (1978), as this characterization will be helpful in the analysis of Algorithm 1.

Lemma 1. *(Nemhauser et al. 1978, Proposition 2.2) Let I be a finite set. Then $f \colon 2^I \to \mathbb{R}$ is a monotone increasing submodular function if and only if*

$$f(T) \leq f(S) + \sum_{i \in T \setminus S} f(i|S), \ \textit{for all } S, T \subseteq I.$$

Now, we demonstrate a statement analogous to Theorem 2 for Algorithm 1. Specifically, we show that the value of the set R, constructed by Algorithm 1, when applied to an instance (I, f, w, B) and an approximation parameter α, approximates the optimal solution of any instance (I, f, w, D), with $D \geq B$, within a factor of $1 - e^{-\frac{B}{\alpha D}}$ if the weight of R is equal to B.

Theorem 3. *Let (I, f, w, B) an* SMKC*-instance and $\alpha \geq 1$ an approximation parameter. Further, let R with $w(R) = B$ be the set constructed in the while-loop of Algorithm 1. Then, for any knapsack capacity $D \geq B$, we have*

$$\frac{f(R)}{\mathrm{Opt}(I, f, w, D)} \geq 1 - e^{-\frac{B}{\alpha D}}.$$

Proof. Let $R = \{R_1, \ldots, R_m\}$, $m < |I|$, and let R_{m+1} be the first item that exceeds the knapsack capacity in Line 5 of Algorithm 1.

By Line 4, we have for all $0 \leq j \leq m$ and any $i \in I \setminus R_{\leq j}$ that

$$\alpha \frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} \geq \max\left\{ \frac{f(e|R_{\leq j})}{w_e} \,\middle|\, e \in I \setminus R_{\leq j} \right\} \geq \frac{f(i|R_{\leq j})}{w_i}. \tag{2}$$

Let $T \subseteq I$, be an optimal solution to an SMKC-instance (I, f, w, D) with $D \geq B$. Then, it follows by Lemma 1 that, for every $0 \leq j \leq m - 1$,

$$\begin{aligned} \mathrm{Opt}(I, f, w, D) = f(T) &\leq f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} f(i|R_{\leq j}) \\ &= f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} \frac{f(i|R_{\leq j})}{w_i} w_i \underset{\text{Ineq. (2)}}{\leq} f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} \alpha \frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} w_i \end{aligned}$$

$$\leq f(R_{\leq j}) + \alpha \frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} \sum_{i \in T \setminus R_{\leq j}} w_i \leq f(R_{\leq j}) + \frac{\alpha D}{w_{R_{j+1}}} f(R_{j+1}|R_{\leq j})$$
$$= f(R_{\leq j}) + \frac{\alpha D}{w_{R_{j+1}}} (f(R_{\leq j+1}) - f(R_{\leq j})).$$

Rearranging the above inequality yields

$$\mathrm{Opt}(I, f, w, D) - f(R_{\leq j+1}) \leq \left(1 - \frac{w_{R_{j+1}}}{\alpha D}\right) (\mathrm{Opt}(I, f, w, D) - f(R_{\leq j}))$$

for every $0 \leq j \leq m - 1$, which implies

$$\mathrm{Opt}(I, f, w, D) - f(R_{\leq j+1}) \leq e^{-\frac{w_{R_{j+1}}}{\alpha D}} (\mathrm{Opt}(I, f, w, D) - f(R_{\leq j})),$$

since $1 - x \leq e^{-x}$ for any $x \geq 0$.

By recursion and $w(R) = B$, it follows

$$\begin{aligned}
\mathrm{Opt}(I, f, w, D) - f(R) &= \mathrm{Opt}(I, f, w, D) - f(R_{\leq m}) \\
&\leq e^{-\frac{w_{R_m}}{\alpha D}} (\mathrm{Opt}(I, f, w, D) - f(R_{\leq m-1})) \\
&\leq e^{-\frac{w_{R_m}}{\alpha D}} e^{-\frac{w_{R_{m-1}}}{\alpha D}} (\mathrm{Opt}(I, f, w, D) - f(R_{\leq m-2})) \\
&\leq e^{-\frac{\sum_{i=1}^{m} w_{R_i}}{\alpha D}} \mathrm{Opt}(I, f, w, D) \\
&= e^{-\frac{B}{\alpha D}} \mathrm{Opt}(I, f, w, D),
\end{aligned}$$

which directly implies $\frac{f(R)}{\mathrm{Opt}(I,f,w,D)} \geq 1 - e^{-\frac{B}{\alpha D}}$. □

Notice that Theorem 3 follows quite similarly to Theorem 2, and that both results coincide when $\alpha = 1$ in Theorem 3.

Now, we are ready to prove an analog of Theorem 1 for Algorithm 1. Specifically, we demonstrate that Algorithm 1 achieves a constant approximation factor depending solely on the approximation parameter α.

Theorem 4. *Algorithm 1 has an approximation factor of at least*

$$1 - e^{-\frac{\gamma}{\alpha}},$$

where γ is the unique solution to $e^{\frac{x}{\alpha}} = 1 + \frac{1-x}{\alpha}$, $x \in [0, 1]$.

Proof. To prove the claim, we first derive two independent lower bounds on the approximation guarantee achievable by Algorithm 1, which we then combine into a single lower bound on the approximation factor of Algorithm 1. To this end, let (I, f, w, B) be an SMKC-instance and $\alpha \geq 1$ an approximation parameter. Further, let $R = \{R_1, \ldots, R_m\}$, $m < |I|$ be the set constructed in the while-loop of Algorithm 1 and let R_{m+1} be the first item that exceeds the knapsack capacity in Line 5 of Algorithm 1.

To derive the lower bounds on the achievable approximation guarantee, we need some auxiliary results regarding the item R_{m+1}. Choose $x \in [0,1]$ such that the equation $\sum_{j=1}^{m} w_{R_j} = xB$ is satisfied. Then,

$$w_{R_{m+1}} \geq (1-x)B, \tag{3}$$

and from submodularity and normality of f, it follows that

$$f(R_{m+1}|R) \leq f(R_{m+1}|\emptyset) = f(R_{m+1}). \tag{4}$$

Furthermore, Inequality (2) from the proof of Theorem 3 holds as well.

Now, we can derive the first lower bound: Let $T \subseteq I$ be an optimal solution to (I, f, w, B). Then we have

$$\begin{aligned}
\mathrm{Opt}(I,f,w,B) = f(T) &\underset{\text{Lem. 1}}{\leq} f(R) + \sum_{i \in T\setminus R} f(i|R) = f(R) + \sum_{i\in T\setminus R} \frac{f(i|R)}{w_i} w_i \\
&\underset{\text{Ineq. (2)}}{\leq} f(R) + \sum_{i\in T\setminus R} \alpha \frac{f(R_{m+1}|R)}{w_{R_{m+1}}} w_i \leq f(R) + \alpha \frac{f(R_{m+1}|R)}{w_{R_{m+1}}} \sum_{i \in T\setminus R} w_i \\
&\leq f(R) + \frac{\alpha B}{w_{R_{m+1}}} f(R_{m+1}|R) \underset{\text{Ineq. (3)}}{\leq} f(R) + \frac{\alpha}{1-x} f(R_{m+1}|R) \\
&\underset{\text{Ineq. (4)}}{\leq} f(R) + \frac{\alpha}{1-x} f(R_{m+1}) \leq \frac{1-x+\alpha}{1-x} \max\{f(R), f(R_{m+1})\} \\
&= \frac{1-x+\alpha}{1-x} \Phi(I,f,w,B,\alpha).
\end{aligned}$$

The second lower bound follows directly from Theorem 3:

$$\Phi(I,f,w,B,\alpha) \geq f(R) \geq (1 - e^{-\frac{x}{\alpha}})\, \mathrm{Opt}(I,f,w,B).$$

Combining both bounds yields

$$\frac{\Phi(I,f,w,B,\alpha)}{\mathrm{Opt}(I,f,w,B)} \geq \min_{0 \leq x \leq 1} \max \left\{ 1 - e^{-\frac{x}{\alpha}}, \frac{1-x}{1-x+\alpha} \right\} \tag{5}$$

Since $1-e^{-\frac{x}{\alpha}}$ is monotonically increasing in x and $\frac{1-x}{1-x+\alpha}$ is monotonically decreasing in x for $x \in [0,1]$, the maximum on the left-hand side of Inequality (5) is minimized when

$$1 - e^{-\frac{x}{\alpha}} = \frac{1-x}{1-x+\alpha}.$$

By

$$\begin{aligned}
1 - e^{-\frac{x}{\alpha}} = \frac{1-x}{1-x+\alpha} &\iff e^{-\frac{x}{\alpha}} = 1 - \frac{1-x}{1-x+\alpha} \\
\iff e^{-\frac{x}{\alpha}} = \frac{\alpha}{1-x+\alpha} &\iff e^{\frac{x}{\alpha}} = 1 + \frac{1-x}{\alpha},
\end{aligned}$$

it follows directly that $\frac{\Phi(I,f,w,B,\alpha)}{\mathrm{Opt}(I,f,w,B)} \geq 1 - e^{-\frac{\gamma}{\alpha}}$, where γ is the unique solution to $e^{\frac{x}{\alpha}} = 1 + \frac{1-x}{\alpha}$ with $x \in [0,1]$. $\square$

Note that Theorem 4 follows quite similarly to Theorem 1, and illustrates the effect of using an α-approximate incremental oracle instead of an exact incremental oracle on the approximation factor of the greedy algorithm. For $\alpha = 1$, Theorem 4 matches the approximation guarantee of Theorem 1 by Wolsey (1982).

3. A greedy algorithm with an approximate incremental oracle for the selection of the first item.

In this section, we focus on a variant of the greedy algorithm studied by Wolsey (1982) that uses an approximate incremental oracle when selecting the first item and an exact incremental oracle when selecting further items.

Algorithm 2:

```
Input: An SMKC-instance $(I, f, w, B)$ and an approximation parameter $\alpha \geq 1$.
Output: Approximately optimal solution to SMKC.
1  $I \leftarrow I \setminus \{i \in I : w_i > B\}$
2  Set $R \leftarrow \emptyset$
3  while $I \setminus R \neq \emptyset$ do
4      if $R = \emptyset$ then
5          Choose $i^* \in I \setminus R$ with $\frac{f(i^*|R)}{w_{i^*}} \geq \frac{1}{\alpha} \max\left\{\frac{f(i|R)}{w_i} \middle| i \in I \setminus R\right\}$ using an
           $\alpha$-approximate incremental oracle
6      else
7          Choose $i^* \in I \setminus R$ with $\frac{f(i^*|R)}{w_{i^*}} \geq \max\left\{\frac{f(i|R)}{w_i} \middle| i \in I \setminus R\right\}$ using an exact
           incremental oracle
8      if $w(R \cup \{i^*\}) \leq B$ then Set $R \leftarrow R \cup \{i^*\}$
9      else break
10 return $\arg\max\{f(R), f(i^*)\}$
```

As before, we consider the items of the set R, constructed by the greedy variant in the order they were added to R. To denote the return of Algorithm 2 applied to an SMKC-instance (I, f, w, B) and an approximation parameter $\alpha \geq 1$, we use $\Phi(I, f, w, B, \alpha)$.

We first demonstrate an analog of Theorem 2 for Algorithm 2. We show that the value of the set R, constructed by Algorithm 2 applied to an instance (I, f, w, B) and an approximation parameter α, approximates the optimal solution of any instance (I, f, w, D) with $D \geq B$ by a constant factor depending on α and the ratio of the weight of the first item added to R and B, if the total weight of R is equal to B.

Theorem 5. *Let (I, f, w, B) be an SMKC-instance and $\alpha \geq 1$ be an approximation parameter. Let R with $w(R) = B$ be the set constructed in the while-loop of Algorithm 2, and let $\lambda := \frac{w_{R_1}}{w(R)}$. Then, for any knapsack capacity $D \geq B$, we have*

$$\frac{f(R)}{\mathrm{Opt}(I, f, w, D)} \geq 1 - \left(1 - \frac{\lambda B}{\alpha D}\right) e^{-(1-\lambda)\frac{B}{D}}.$$

Proof. Let $R = \{R_1, \dots, R_m\}$, $m < |I|$ and let R_{m+1} be the first item that exceeds the knapsack capacity in Line 8 of Algorithm 2.

Let $T \subseteq I$ be an optimal solution to an instance (I, f, w, D) with $B \leq D$. Since Algorithm 2 selects the item R_1 with an α-incremental oracle, we obtain

$$\begin{aligned}\mathrm{Opt}(I, f, w, D) = f(T) &\underset{\text{Lem. 1}}{\leq} f(R_{\leq 0}) + \sum_{i \in T \setminus R_{\leq 0}} f(i|R_{\leq 0}) = \sum_{i \in T} \frac{f(i|\emptyset)}{w_i} w_i \\ &\leq \sum_{i \in T} \alpha \frac{f(R_1|\emptyset)}{w_{R_1}} w_i = \alpha \frac{f(R_1)}{w_{R_1}} \sum_{i \in T} w_i \leq \frac{\alpha D}{w_{R_1}} f(R_1),\end{aligned}$$

which implies

$$\mathrm{Opt}(I, f, w, D) - f(R_1) \leq \left(1 - \frac{w_{R_1}}{\alpha D}\right) \mathrm{Opt}(I, f, w, D).$$

All other items are selected using an exact incremental oracle. Hence, we have for all $1 \leq j \leq m$ and any $i \in I \setminus R_{\leq j}$ that

$$\frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} \geq \max \left\{ \frac{f(e|R_{\leq j})}{w_e} \middle| e \in I \setminus R_{\leq j} \right\} \geq \frac{f(i|R_{\leq j})}{w_i}, \tag{6}$$

and by Lemma 1 it follows, for every $1 \leq j \leq m - 1$, that

$$\begin{aligned}\mathrm{Opt}(I, f, w, D) = f(T) \leq f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} f(i|R_{\leq j}) &= f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} \frac{f(i|R_{\leq j})}{w_i} w_i \\ \leq f(R_{\leq j}) + \sum_{i \in T \setminus R_{\leq j}} \frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} w_i &= f(R_{\leq j}) + \frac{f(R_{j+1}|R_{\leq j})}{w_{R_{j+1}}} \sum_{i \in T \setminus R_{\leq j}} w_i \\ \leq f(R_{\leq j}) + \frac{D}{w_{R_{j+1}}} f(R_{j+1}|R_{\leq j}) &= f(R_{\leq j}) + \frac{D}{w_{R_{j+1}}} (f(R_{\leq j+1}) - f(R_{\leq j})),\end{aligned}$$

Rearranging the above inequality yields, for every $1 \leq j \leq m - 1$,

$$\mathrm{Opt}(I, f, w, D) - f(R_{\leq j+1}) \leq \left(1 - \frac{w_{R_{j+1}}}{D}\right) (\mathrm{Opt}(I, f, w, D) - f(R_{\leq j})).$$

By recursion, we directly obtain

$$\begin{aligned}\mathrm{Opt}(I, f, w, D) - f(R) &= \mathrm{Opt}(I, f, w, D) - f(R_{\leq m}) \\ &\leq \prod_{j=2}^{m} \left(1 - \frac{w_{R_j}}{D}\right) (\mathrm{Opt}(I, f, w, D) - f(R_{\leq 1})) \\ &\leq (1 - \frac{w_{R_1}}{\alpha D}) \prod_{j=2}^{m} \left(1 - \frac{w_{R_j}}{D}\right) \mathrm{Opt}(I, f, w, D) \\ &\leq (1 - \frac{w_{R_1}}{\alpha D}) e^{-\frac{\sum_{j=2}^{m} w_{R_j}}{D}} \mathrm{Opt}(I, f, w, D)\end{aligned}$$

$$= (1 - \frac{w_{R_1}}{\alpha D})e^{-\frac{w(R)-w_{R_1}}{D}} \operatorname{Opt}(I, f, w, D)$$
$$= (1 - \frac{\lambda B}{\alpha D})e^{-(1-\lambda)\frac{B}{D}} \operatorname{Opt}(I, f, w, D),$$

where the last inequality follows by $1 - x \le e^{-x}$ for $x \ge 0$.

The claim follows by

$$\begin{aligned}
&\operatorname{Opt}(I, f, w, D) - f(R) \le (1 - \frac{\lambda B}{\alpha D})e^{-(1-\lambda)\frac{B}{D}} \operatorname{Opt}(I, f, w, D) \\
\iff\ & f(R) - \operatorname{Opt}(I, f, w, D) \ge -(1 - \frac{\lambda B}{\alpha D})e^{-(1-\lambda)\frac{B}{D}} \operatorname{Opt}(I, f, w, D) \\
\iff\ & f(R) \ge (1 - (1 - \frac{\lambda B}{\alpha D})e^{-(1-\lambda)\frac{B}{D}}) \operatorname{Opt}(I, f, w, D). \qquad \square
\end{aligned}$$

Next, we present an approximation factor for Algorithm 2. In contrast to the approximation factor of Algorithm 1, we have so far not been able to express this approximation factor as a closed-form. Instead, it is defined as the optimal value of a minimization problem that depends on α. Nevertheless, numerical methods, such as SLSQP (Sequential Least Squares Programming), can be used to solve this problem effectively.

Theorem 6. *Algorithm 2 has an approximation factor of at least*

$$\min\left\{\frac{1-x}{2-x} : 2 - x = \frac{\alpha}{\alpha - \lambda x}e^{(1-\lambda)x},\ \lambda \in (0, 1]\right\}. \tag{7}$$

Proof. Similar to the proof of Theorem 4, we prove the claim by first deriving two independent lower bounds on the approximation guarantee achievable by Algorithm 2, which we then combine into one lower bound on the approximation factor of Algorithm 2. Let (I, f, w, B) be an SMKC-instance and $\alpha \ge 1$ an approximation parameter. Further, let $R = \{R_1, \ldots, R_m\}$, $m < |I|$ be the set constructed in the while-loop of Algorithm 2 and let R_{m+1} be the first item that exceeds the knapsack capacity in Line 8 of Algorithm 2.

To derive the lower bounds on the approximation guarantee, we use the following auxiliary results: Choose $x \in [0, 1]$ such that the equation $\sum_{j=1}^{m} w_{R_j} = xB$ is satisfied. Then Inequality (3) and Inequality (4) from the proof of Theorem 4 are also valid here.

Now, we can derive the first lower bound: Let $T \subseteq I$ be an optimal solution to (I, f, w, B). By Lemma 1, we have

$$\begin{aligned}
\operatorname{Opt}(I, f, w, B) = f(T) &\le f(R) + \sum_{i \in T\setminus R} f(i|R) = f(R) + \sum_{i \in T\setminus R} \frac{f(i|R)}{w_i} w_i \\
&\le f(R) + \sum_{i \in T\setminus R} \frac{f(R_{m+1}|R)}{w_{R_{m+1}}} w_i = f(R) + \frac{f(R_{m+1}|R)}{w_{R_{m+1}}} \sum_{i \in T\setminus R} w_i \\
&\le f(R) + \frac{B}{w_{R_{m+1}}} f(R_{m+1}|R) \underset{\text{Ineq. (3)}}{\le} f(R) + \frac{1}{1-x} f(R_{m+1}|R) \\
&\underset{\text{Ineq. (4)}}{\le} f(R) + \frac{1}{1-x} f(R_{m+1}) \le \frac{2-x}{1-x} \max\{f(R), f(R_{m+1})\}
\end{aligned}$$

$$= \frac{2-x}{1-x}\Phi(I,f,w,B,\alpha),$$

where the second inequality follows from Line 7 of Algorithm 2. By Theorem 5, we obtain the second bound:

$$\Phi(I,f,w,B,\alpha) \geq f(R) \geq \left(1-\left(1-\frac{\lambda}{\alpha}x\right)\right)e^{-(1-\lambda)x}\,\mathrm{Opt}(I,f,w,B), \text{ where } \lambda = \frac{w_{R_1}}{w(R)}.$$

Combining both bounds yields

$$\frac{\Phi(I,f,w,B,\alpha)}{\mathrm{Opt}(I,f,w,B)} \geq \min_{\substack{0\leq x\leq 1\\ 0<\lambda\leq 1}} \max\left\{1-\left(1-\frac{\lambda}{\alpha}x\right)e^{-(1-\lambda)x}, \frac{1-x}{2-x}\right\} \tag{8}$$

Since $1-\left(1-\frac{\lambda}{\alpha}x\right)e^{-(1-\lambda)x}$ increases monotonically in x and $\frac{1-x}{2-x}$ decreases monotonically in x for $x \in [0,1]$, the maximum on the left-hand side of Inequality (8) is minimized when

$$1-\left(1-\frac{\lambda}{\alpha}x\right)e^{-(1-\lambda)x} = \frac{1-x}{2-x}.$$

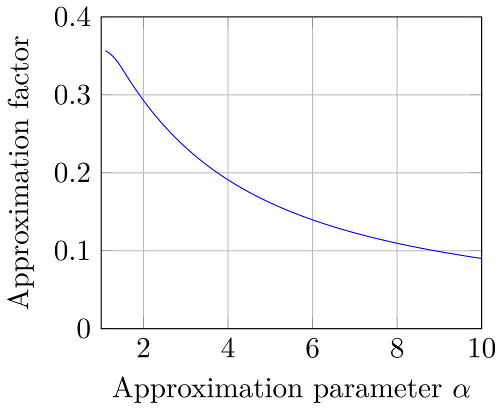
Hence, it follows by

$$\begin{aligned} 1-\left(1-\frac{\lambda}{\alpha}x\right)e^{-(1-\lambda)x} = \frac{1-x}{2-x} &\iff \left(1-\frac{\lambda}{\alpha}x\right)e^{-(1-\lambda)x} = \frac{1}{2-x} \\ &\iff \frac{\alpha}{\alpha-\lambda x}e^{(1-\lambda)x} = 2-x, \end{aligned}$$

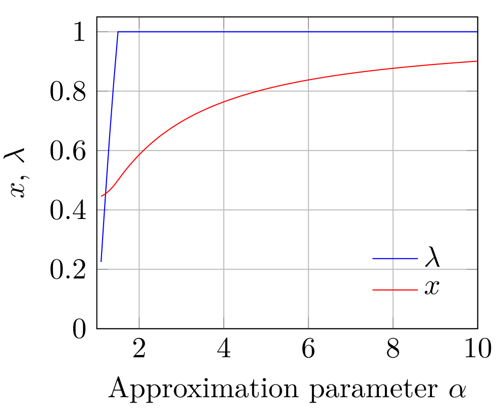
that

$$\frac{\Phi(I,f,w,B,\alpha)}{\mathrm{Opt}(I,f,w,B)} \geq \min\left\{\frac{1-x}{2-x} : 2-x = \frac{\alpha}{\alpha-\lambda x}e^{(1-\lambda)x},\ \lambda \in (0,1]\right\}. \qquad \square$$

To illustrate the effect of the approximation parameter α on the approximation factor of Algorithm 2, we solved the minimization problem (7) for different values of α. Furthermore, for each approximation parameter α considered, we present the minimizer λ of Problem (7), and the corresponding value of x in (7). In this minimization problem, x represents the ratio between the knapsack capacity B of an instance (I,f,w,B) and the total weight $w(R)$ of the set R constructed by Algorithm 2 when Algorithm 2 matches the approximation factor for that instance. The variable λ represents the ratio between the weight of the first item added to R and the total weight of R, when Algorithm 2 matches the approximation factor.



(a) Approximation factor of Algorithm 2 in dependence of the approximation parameter α.



(b) λ and x from Problem (7) in dependence of the approximation parameter α.

Figure (a) illustrates that the approximation factor of Algorithm 2 decreases as the approximation parameter α increases. Specifically, permitting the selection of an item whose relative marginal gain is substantially lower than the maximum relative marginal gain in the first iteration leads to a significantly lower approximation factor.

Figure (b) shows that, in instances where Algorithm 2 returns a solution such that the approximation factor is matched, the proportion of the first item in the total weight of the packed knapsack R also increases with the approximation parameter α. Already for $\alpha = 1.6$, the knapsack set R packed by Algorithm 2 consists solely of the item selected in the first iteration of the while loop, as the adverse effect of “high” approximation parameters outweighs the negative impact of submodularity on the approximation factor. In addition, Figure (b) demonstrates that, for such instances, the ratio of the weight of the packed knapsack R to the knapsack capacity B increases as the approximation parameter α increases.

4. Conclusion.

We extended the classic greedy algorithm in two ways to utilize approximate incremental oracles and established approximation guarantees for both variants.

It remains an open question whether the approximation factor of the greedy variant, using an approximate incremental oracle in the first iteration and an exact incremental oracle in all subsequent iterations, can be expressed as a *succinct* closed formula.

Another closely related question is to determine an approximation factor in settings where a learning effect occurs across the iterations of the greedy algorithm, i.e., the oracle improves incrementally in each iteration.

Acknowledgements.

The authors thank the DFG for their support within RTG 2126 “Algorithmic Optimization”.

References

Ageev, A.A. and M.I. Sviridenko (1999). “An 0.828-approximation algorithm for the uncapacitated facility location problem.” In: *Discrete Applied Mathematics* 93.2-3, pp. 149–156. DOI: 10.1016/S0166-218X(99)00103-1.

Cornuejols, Gerard, Marshall L. Fisher, and George L. Nemhauser (1977). “Exceptional Paper - Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms.” In: *Management science* 23.8, pp. 789–810. DOI: 10.1287/mnsc.23.8.789.

Feige, Uriel (1998). “A threshold of ln n for approximating set cover.” In: *Journal of the ACM (JACM)* 45.4, pp. 634–652. DOI: 10.1145/285055.285059.

Feige, Uriel and Jan Vondrák (2006). “Approximation algorithms for allocation problems: Improving the factor of 1-1/e.” In: *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, pp. 667–676. DOI: 10.1109/FOCS.2006.14.

Fisher, M. L., G. L. Nemhauser, and L. A. Wolsey (1978). “An analysis of approximations for maximizing submodular set functions-II.” In: *Polyhedral Combinatorics: Dedicated to the memory of D.R. Fulkerson*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 73–87. DOI: 10.1007/BFb0121195.

Fleischer, Lisa, Michel X. Goemans, Vahab S. Mirrokni, and Maxim Sviridenko (2006). “Tight Approximation Algorithms for Maximum General Assignment Problems.” In: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*. SODA '06. Miami, Florida: Society for Industrial and Applied Mathematics, pp. 611–620.

Goemans, Michel X. and David P. Williamson (1995). “Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming.” In: *Journal of the ACM (JACM)* 42.6, pp. 1115–1145. DOI: 10.1145/227683.227684.

Goundan, Pranava R. and Andreas S. Schulz (2007). “Revisiting the Greedy Approach to Submodular Set Function Maximization.” In: *Working Paper, Massachusetts Institute of Technology*.

Kempe, David, Jon Kleinberg, and Éva Tardos (2015). “Maximizing the Spread of Influence through a Social Network.” In: *Theory of Computing* 11.4, pp. 105–147. DOI: 10.4086/toc.2015.v011a004.

Krause, Andreas and Carlos Guestrin (2007). “Near-optimal observation selection using submodular functions.” In: *Proceedings of the 22nd national conference on Artificial intelligence-Volume 2*. AAAI'07, pp. 1650–1654.

Krause, Andreas, H. Brendan McMahan, Carlos Guestrin, and Anupam Gupta (2008). "Robust Submodular Observation Selection." In: *Journal of Machine Learning Research* 9.93, pp. 2761–2801.

Lin, Hui and Jeff Bilmes (2011). "A class of submodular functions for document summarization." In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. HLT '11. Association for Computational Linguistics, pp. 510–520.

Nemhauser, G. L., L. A. Wolsey, and M. L. Fisher (1978). "An analysis of approximations for maximizing submodular set functions-I." In: *Mathematical Programming* 14, pp. 265–294. DOI: 10.1007/BF01588971.

Sahni, Sartaj (1975). "Approximate algorithms for the 0/1 knapsack problem." In: *Journal of the ACM, (JACM)* 22.1, pp. 115–124. DOI: 10.1145/321864.321873.

Sviridenko, Maxim (2004). "A note on maximizing a submodular set function subject to a knapsack constraint." In: *Operations Research Letters* 32.1, pp. 41–43. DOI: 10.1016/S0167-6377(03)00062-2.

Tschiatschek, Sebastian, Rishabh Iyer, Haochen Wei, and Jeff Bilmes (2014). "Learning mixtures of submodular functions for image collection summarization." In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'14, pp. 1413–1421.

Wei, Kai, Rishabh Iyer, and Jeff Bilmes (2015). "Submodularity in data subset selection and active learning." In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. PMLR, pp. 1954–1963.

Wolsey, Laurence A. (1982). "Maximising Real-Valued Submodular Functions: Primal and Dual Heuristics for Location Problems." In: *Mathematics of Operations Research* 7.3, pp. 410–425. DOI: 10.1287/moor.7.3.410.