The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Pareto Optimization for Subset Selection with Dynamic Cost Constraints

Vahid Roostapour,[1] Aneta Neumann,[1] Frank Neumann,[1] Tobias Friedrich[1, 2]
[1]Optimisation and Logistics, School of Computer Science, The University of Adelaide, Adelaide, Australia
[2]Chair of Algorithm Engineering, Hasso Plattner Institute, Potsdam, Germany
{vahid.roostapour, aneta.neumann, frank.neumann}@adelaide.edu.au, tobias.friedrich@hpi.de

Abstract

In this paper, we consider the subset selection problem for function f with constraint bound B which changes over time. We point out that adaptive variants of greedy approaches commonly used in the area of submodular optimization are not able to maintain their approximation quality. Investigating the recently introduced POMC Pareto optimization approach, we show that this algorithm efficiently computes a $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$-approximation, where α_f is the submodularity ratio of f, for each possible constraint bound $b \leq B$. Furthermore, we show that POMC is able to adapt its set of solutions quickly in the case that B increases. Our experimental investigations for the influence maximization in social networks show the advantage of POMC over generalized greedy algorithms.

Introduction

Many combinatorial optimization problems consist of optimizing a given function under a given constraint. Constraints often limit the resources available to solve a given problem and may change over time. For example, in the area of supply chain management, the problem may be constrained by the number of vehicles available which may change due to vehicle failures and vehicles being repaired or added as a new resource.

Submodular functions form an important class of problems as many important optimization problems can be modeled by them. The area of submodular function optimization under given static constraints has been studied quite extensively in the literature (Nemhauser, Wolsey, and Fisher 1978; Lee et al. 2010; Lee, Sviridenko, and Vondrák 2010; Krause and Golovin 2014). In the case of monotone submodular functions, greedy algorithms are often able to achieve the best possible worst case approximation guarantee (unless P=NP).

Recently, Pareto optimization approaches have been investigated for a wide range of subset selection problems (Friedrich and Neumann 2015; Qian, Yu, and Zhou 2015; Qian et al. 2017a; 2017b). It has been shown in (Qian et al. 2017b) that an algorithm called POMC is able to achieve a $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$-approximation where α_f measures the closeness of the considered function f to submodularity. The approximation matches the worst-case performance ratio of the generalized greedy algorithm (Zhang and Vorobeychik 2016).

Our contribution

In this paper, we study monotone functions with a constraint where the constraint bound B changes over time. Such constraint changes reflect real-world scenarios where resources vary during the process. We show that greedy algorithms have difficulties in adapting their current solutions after changes have happened. In particular, we show that there are simple dynamic versions of the classical knapsack problem where adding elements in a greedy fashion when the constraint bound increases over time can lead to an arbitrary bad performance ratio. For the case where constraint bounds decrease over time, we introduce a submodular graph covering problem and show that the considered adaptive generalized greedy algorithm may encounter an arbitrarily bad performance ratio on this problem.

Investigating POMC, we show that this algorithm obtains for each constraint bound $b \in [0, B]$, a $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$-approximation efficiently. Furthermore, when relaxing the bound B to $B^* > B$, ϕ-approximations for all values of $b \in [0, B^*]$ are obtained efficiently. We evaluate the considered algorithms on influence maximization in social networks that have dynamic routing or cardinality constraints. Benchmarking the algorithms over sequences of dynamic changes, we show that POMC obtains superior results to the generalized greedy and adaptive generalized greedy algorithms. Furthermore, POMC shows an even more prominent advantage as more dynamic changes are carried out because it is able to improve the quality of its solutions and cater for changes that occur in the future.

The paper is structured as follows. In the next section, we define the dynamic problems and the algorithms under investigation. Then, we show that the adaptive generalized greedy algorithm may encounter an arbitrary bad performance ratio even when starting with an optimal solution for the initial budget. In contrast, we show that POMC is able to maintain a ϕ-approximation efficiently. We present experimental investigations for influence maximization in social networks that show the advantage of the Pareto optimization approach and finish with some concluding remarks.

Problem Formulation and Algorithms

In this paper we consider optimization problems in which the cost function and objective functions are monotone and quantified according to their closeness to submodularity.

Different definitions are given for submodularity (Nemhauser, Wolsey, and Fisher 1978) and we use the following one in this paper. For a given set $V = \{v_1, \cdots, v_n\}$, a function $f : 2^V \to \mathbb{R}$ is submodular if

for any $X \subseteq Y \subseteq V$ and $v \notin Y$

$$f(X \cup v) - f(X) \geq f(Y \cup v) - f(Y). \quad (1)$$

In addition, we consider how much a function is close to being submodular, measured by the submodularity ratio (Zhang and Vorobeychik 2016). The function f is α_f-submodular where

$$\alpha_f = \min_{X \subseteq Y, v \notin Y} \frac{f(X \cup v) - f(X)}{f(Y \cup v) - f(Y)}.$$

This definition is equivalent to the Equation 1 when $\alpha_f = 1$, i.e. we have submodularity in this case. Another notion which is used in our analysis is the curvature of function f. The curvature measures the deviation from linearity and reflects the effect of marginal contribution according to the function f (Conforti and Cornuéjols 1984; Vondrák 2010). For a monotone submodular function $f : 2^V \to \mathbb{R}^+$,

$$\kappa_f = 1 - \min_{v \in V} \frac{f(V) - f(V \setminus v)}{f(v)}$$

is defined as the total curvature of f.

In many applications the function to be optimized f comes with a cost function c which is subject to a given cost constraint B. Often the cost function c cannot be evaluated precisely. Hence, the function $\hat{c}$ which is ψ-approximation of c is used (Zhang and Vorobeychik 2016). Moreover, according to the submodularity of f, the aim is to find a good approximation instead of finding the optimal solution.

Consider the static version of an optimization problem defined in (Qian et al. 2017b).

Definition 1 (The Static Problem)**.** *Given a monotone objective function $f : 2^V \to \mathbb{R}^+$, a monotone cost function $c : 2^V \to \mathbb{R}^+$ and a budget B, the goal is to compute a solution X such that*

$$X = \arg\max_{Y \subseteq V} f(Y) \text{ s.t. } c(Y) \leq B.$$

As for the static case investigated in (Qian et al. 2017b), we are interested in a ϕ-approximation where $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$ depends on the submodularity ratio.

Zhang and Vorobeychik considered the performance of the generalized greedy algorithm (Zhang and Vorobeychik 2016), given in Algorithm 1, according to the approximated cost function $\hat{c}$. Starting from the empty set, the algorithm always adds the element with the largest objective to cost ratio that does not violate the given constraint bound B.

Let $K_c = \max\{|X| : c(X) \leq B\}$. The optimal solution $\tilde{X}_B$ in these investigations is defined as $\tilde{X}_B = \arg\max\{f(X) \mid c(X) \leq \alpha_c \frac{B(1+\alpha_c^2(K_c-1)(1-\kappa_c)))}{\psi K_c}\}$ where α_c is the submodularity ratio of c. This formulation gives the value of an optimal solution for a slightly smaller budget constraint. The goal is to obtain a good approximation of $f(\tilde{X}_B)$ in this case.

It has been shown in (Zhang and Vorobeychik 2016) that the generalized greedy algorithm, which adds the item with the highest marginal contribution to the current solution in each step, achieves a $(1/2)(1-\frac{1}{e})$-approximate solution if f is monotone and submodular. (Qian et al. 2017b) extended these results to objective functions with α_f submodularity ratio and proved that the generalized greedy algorithm obtains a $\phi = (\alpha_f/2)(1-\frac{1}{e^{\alpha_f}})$-approximation. For the remainder of this paper, we assume $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$ and are interested in obtaining solutions that are ϕ-approximation for the considered problems.

In this paper, we study the dynamic version of problem given in Definition 1.

Definition 2 (The Dynamic Problem)**.** *Let X be a ϕ-approximation for the problem in Definition 1. The dynamic problem is given by a sequence of changes where in each change the current budget B changes to $B^* = B + d$, $d \in \mathbb{R}_{\geq -B}$. The goal is to compute a ϕ-approximation X' for each newly given budget B^*.*

The Dynamic Problem evolves over time by the changing budget constraint bounds. Note that every fixed constraint bound gives a static problem and any good approximation algorithm can be run from scratch for the newly given budget. However, the main focus of this paper are algorithms that can adapt to changes of the constraint bound.

Algorithms

We consider dynamic problems according to Definition 2 with $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$ and are interested in algorithms that adapt their solutions to the new constraint bound B^* and obtain a ϕ-approximation for the new bound B^*. As the generalized greedy algorithm can be applied to any bound B, the first approach would be to run it for the newly given bound B^*. However, this might lead to totally different solutions and adaptation of already obtained solutions might be more beneficial. Furthermore, adaptive approaches that change the solution based on the constraint changes are of interest as they might be faster in obtaining such solutions and/or be able to learn good solutions for the different constraint bounds that occur over time.

Based on the generalized greedy algorithm, we introduce the adaptive generalized greedy algorithm. This algorithm is modified from Algorithm 1 in a way that enables it to deal with a dynamic change. Let X be the current solution of the algorithm. When a dynamic change decreases the budget constraint, the algorithm removes items from X according to their marginal contribution, until it achieves a feasible solution. When there is a dynamic increase, this algorithm behaves similarly to the generalized greedy algorithm.

Furthermore, we consider the Pareto optimization approach POMC (Algorithm 3) which is also known as Global SEMO in the evolutionary computation literature (Laumanns, Thiele, and Zitzler 2004; Friedrich et al. 2010; Friedrich and Neumann 2015). POMC is a multi-objective

Algorithm 1: Generalized Greedy Algorithm

input: Initial budget constraint B.

$X \leftarrow \emptyset$;
$V' \leftarrow V$;
repeat
  $v^* \leftarrow \arg\max_{v \in V'} \frac{f(X \cup v) - f(X)}{\hat{c}(X \cup v) - \hat{c}(X)}$;
  if $\hat{c}(X \cup v^*) \leq B$ **then**
    $X \leftarrow X \cup v^*$;
  $V' \leftarrow V' \setminus \{v^*\}$;
until $V' \leftarrow \emptyset$;
$v^* \leftarrow \arg\max_{v \in V; \hat{c}(v) \leq B} f(v)$;
return $\arg\max_{S \in \{X, v^*\}} f(S)$;

Algorithm 2: Adaptive Generalized Greedy Algorithm

input: Initial solution X, Budget constraint B, New budget constraint B^*.

if $B^* < B$ **then**
  while $\hat{c}(X) > B^*$ **do**
    $v^* \leftarrow \arg\min_{v \in X} \frac{f(X) - f(X \setminus \{v\})}{\hat{c}(X) - \hat{c}(X \setminus \{v\})}$;
    $X \leftarrow X \setminus \{v^*\}$;
else if $B^* > B$ **then**
  $V' \leftarrow V \setminus X$;
  repeat
    $v^* \leftarrow \arg\max_{v \in V'} \frac{f(X \cup v) - f(X)}{\hat{c}(X \cup v) - \hat{c}(X)}$;
    if $\hat{c}(X \cup v^*) \leq B^*$ **then**
      $X \leftarrow X \cup v^*$;
    $V' \leftarrow V' \setminus \{v^*\}$;
  until $V' \leftarrow \emptyset$;
$v^* \leftarrow \arg\max_{v \in V; \hat{c}(v) \leq B^*} f(v)$;
return $\arg\max_{S \in \{X, v^*\}} f(S)$;

optimization approach which is proven to perform better than the generalized greedy algorithm in case of local optima (Qian et al. 2017b). We reformulate the problem as a bi-objective problem in order to use POMC as follows:

$$\arg\max X \in \{0,1\}^n (f_1(X), f_2(X)),$$

where:

$$f_1(X) = \begin{cases} -\infty, & \hat{c}(X) > B + 1 \\ f(X), & \text{otherwise} \end{cases}, f_2(X) = -\hat{c}(X).$$

This algorithm optimizes the cost function and the objective function simultaneously. To achieve this, it uses the concept of dominance to compare two solutions. Solution X_1 dominates X_2, denoted by $X_1 \succeq X_2$, if $f_1(X_1) \geq f_1(X_2) \wedge f_2(X_1) \geq f_2(X_2)$. The dominance is strict, $\succ$, when at least one of the inequalities is strict. POMC produces a population of non-dominated solutions and optimizes them during the optimization process. In each iteration, it chooses solution X randomly from the population and flips each bit of the solution with the probability of $1/n$. It adds the mutated solution X' to the population only if

Algorithm 3: POMC Algorithm

input: Initial budget constraint B, time T

$X \leftarrow \{0\}^n$;
Compute $(f_1(X), f_2(X))$;
$P \leftarrow \{x\}$;
$t \leftarrow 0$;
while $t < T$ **do**
  Select X from P uniformly at random;
  $X' \leftarrow$ flip each bit of X with probability $\frac{1}{n}$;
  Compute $(f_1(X'), f_2(X'))$;
  if $\nexists Z \in P$ *such that* $Z \succ X'$ **then**
    $P \leftarrow (P \setminus \{Z \in P \mid X' \succeq Z\}) \cup \{X'\}$;
  $t = t + 1$;
return $\arg\max_{X \in P: \hat{c}(X) \leq B} f(x)$

there is no solution in the population that dominates X'. All the solutions which are dominated by X' will be deleted from the population afterward.

Note that we only compute the objective vector $(f_1(X), f_2(X))$ when the solution X is created. This implies that the objective vector is not updated after changes to the constraint bound B. As a consequence solutions whose constraint exceeds the value of $B + 1$ for a newly given bound are kept in the population. However, newly produced individuals exceeding $B + 1$ for the current bound B are not included in the population as they are dominated by the initial search point 0^n. We are using the value $B + 1$ instead of B in the definition of f_1 as this gives the algorithm some look ahead for larger constraint bounds. However, every value of at least B would work for our theoretical analyses. The only drawback would be a potentially larger population size which influences the value $P_{\max}$ in our runtime bounds.

Adaptive Generalized Greedy Algorithm

In this section we analyze the performance of the adaptive generalized greedy algorithm. This algorithm is a modified version of the generalized greedy using the same principle in adding and deleting items. However, in this section we prove that the adaptive generalized greedy algorithm is not able to deal with the dynamic change, i.e., the approximation obtained can become arbitrarily bad during a sequence of dynamic changes.

In order to show that the adaptive generalized greedy algorithm can not deal with dynamic increases of the constraint bound, we consider a special instance of the classical knapsack problem. Note that the knapsack problem is special submodular problem where both the objective and the cost function are linear.

Given $n + 1$ items $e_i = (c_i, f_i)$ with cost c_i and value f_i independent of the choice of the other items, we assume there are items $e_i = (1, \frac{1}{n})$, $1 \leq i \leq n/2$, $e_i = (2, 1)$, $n/2 + 1 \leq i \leq n$, and a special item $e_{n+1} = (1, 3)$. We have $f_{inc}(X) = \sum_{e_i \in X} f_i$ and $c_{inc}(X) = \sum_{e_i \in X} c_i$ as the linear objective and constraint function, respectively.

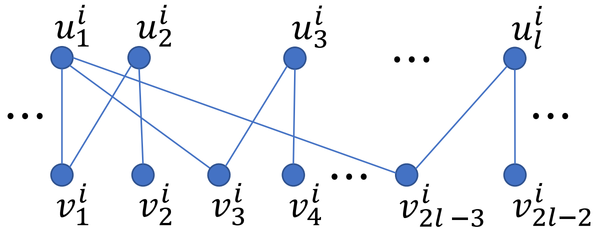

Figure 1: Single subgraph G_i of $G = (U, V, E)$

Theorem 3. *Given the dynamic knapsack problem (f_{inc}, c_{inc}), starting with $B = 1$ and increasing the bound $n/2$ times by 1, the adaptive generalized greedy algorithm computes a solution that has approximation ratio $O(1/n)$.*

Proof. For the constraint $B = 1$ the optimal solution is $\{e_{n+1}\}$. Now let there be $n/2$ dynamic changes where each of them increases B by 1. In each change, the algorithm can only pick an item from $\{e_1, \cdots, e_{n/2}\}$, otherwise it violates the budget constraint. After $n/2$ changes, the budget constraint is $1 + n/2$ and the result of the algorithm is $S = \{e_{n+1}, e_1, \cdots, e_n/2\}$ with $f(S) = 3+(n/2)\cdot(1/n) = 7/2$ and $c(S) = 1 + n/2$. However, an optimal solution for budget $1 + n/2$ is $S^* = \{e_{n+1}, e_{n/2+1}, \ldots, e_{\frac{3n}{4}}\}$ with $f(S^*) = 3 + \frac{n}{4}$. Hence, the approximation ratio in this example is $(7/2)/(3 + n/4) = O(1/n)$ □

Now we consider the case where the constraint bound decreases over time and show that the adaptive generalized greedy algorithm may also encounter situations where the approximation ratio becomes arbitrarily bad over time.

We consider the following *Graph Coverage Problem*. Let $G = (U, V, E)$ be a bipartite graph with bipartition U and V of vertices with $|U| = n$ and $|V| = m$. The goal is to select a subset $S \subseteq U$ with $|S| \leq B$ such that the number of neighbors of S in V is maximized. Note that the objective function $f(S)$ measuring the number of neighbors of S in V is monotone and submodular.

We consider the graph $G = (U, V, E)$ which consists of k disjoint subgraphs

$$G_i = (U_i = \{u_1^i, \cdots, u_l^i\}, V_i = \{v_1^i, \cdots, v_{2l-2}^i\}, E_i)$$

(see Figure 1). Node u_1^i is connected to nodes v_{2j-1}^i, $1 \leq j \leq l-1$. Moreover, each vertex u_j^i, $2 \leq j \leq l$ is connected to two vertices v_{2j-3}^i and v_{2j-2}^i. We assume that $k = \sqrt{n}$ and $l = n/k = \sqrt{n}$.

Theorem 4. *Starting with the optimal set $S = U$ and budget $B = n$, there is a specific sequence of dynamic budget reductions such that the solution obtained by the adaptive generalized greedy algorithm has the approximation ratio $O(1/\sqrt{n})$.*

Proof. Let the adaptive generalized greedy algorithm be initialized with $X = U$ and $B = n = kl$. We assume that the budget decreases from n to k where each single decrease reduces the budget by 1. In the first k steps, to change the cost of solution from n to $n-k$, the algorithm removes the nodes u_1^i, $1 \leq i \leq k$, as they have a marginal contribution of 0. Following these steps, all the remaining nodes have the same marginal contribution of 2. The solution X of size k obtained by the removal steps of the adaptive generalized greedy algorithm contains k vertices which are connected to $2k$ nodes of V, thus $f(X) = 2k = 2\sqrt{n}$. Such a solution is returned by the algorithm for $B = k$ as the most valuable single node has at most $(l-1) = (\sqrt{n}-1)$ neighbors in V. For $B = k$, the optimal solution $X^* = \{u_1^i \mid 1 \leq i \leq k\}$ has $f(X^*) = k(l-1) = n - \sqrt{n}$. Therefore, the approximation ratio achieved by the adaptive generalized greedy algorithm is upper bounded by $(2\sqrt{n})/(n-\sqrt{n}) = O(1/\sqrt{n})$. □

Pareto Optimization

In this section we analyze the behavior of POMC facing a dynamic change. According to Lemma 3 in (Qian et al. 2017b), we have for any $X \subseteq V$ and $v^* = \arg\max_{v \notin X} \frac{f(X \cup v) - f(X)}{\hat{c}(X \cup v) - \hat{c}(X)}$:

$$f(X \cup v^*) - f(X) \geq \alpha_f \frac{\hat{c}(X \cup v^*) - \hat{c}(X)}{B} \cdot (f(\tilde{X}) - f(X)).$$

We denote by $\delta_{\hat{c}} = \min\{\hat{c}(X \cup v) - \hat{c}(X) \mid X \subseteq V, v \notin X\}$ the smallest contribution of an element to the cost of a solution for the given problem. Moreover, let $P_{\max}$ be the maximum size of POMC's population during the optimization process.

The following theorem considers the static case and shows that POMC computes a ϕ-approximation efficiently for every budget $b \in [0, B]$.

Theorem 5. *Starting from $\{0\}^n$, POMC computes, for any budget $b \in [0, B]$, a $\phi = (\alpha_f/2)(1 - 1/e^{\alpha_f})$-approximate solution after $T = cnP_{\max} \cdot \frac{B}{\delta_{\hat{c}}}$ iterations with the constant probability, where $c \geq 8e+1$ is a sufficiently large arbitrary constant.*

Proof. We first consider the number of iterations to find a $(\alpha_f/2)\left(1 - (1 - \frac{\alpha_f}{k})^k\right)$-approximate solution for a budget $b \in [0, B]$ and some k. We consider the largest value of i for which there is a solution X in the population where $\hat{c}(X) \leq i < b$ and

$$f(X) \geq \left(1 - \left(1 - \alpha_f \frac{i}{bk}\right)^k\right) \cdot f(\tilde{X}_b)$$

holds for some k. Initially, it is true for $i = 0$ with $X = \{0\}^n$. We now show that adding v^* to the current solution has the desired contribution to achieve a ϕ-approximate solution. Let $X \subseteq V$ and $v^* = \arg\max_{v \notin X} \frac{f(X \cup v) - f(X)}{\hat{c}(X \cup v) - \hat{c}(X)}$.

Assume that

$$f(X) \geq \left(1 - \left(1 - \alpha_f \frac{i}{bk}\right)^k\right) \cdot f(\tilde{X}_b)$$

holds for some $\hat{c}(X) \leq i < b$ and k. Then adding v^* leads to

$$f(X \cup v^*) \geq$$
$$\left(1 - \left(1 - \alpha_f \frac{i + \hat{c}(X \cup v^*) - \hat{c}(X)}{b(k+1)}\right)^{k+1}\right) \cdot f(\tilde{X}_b).$$

This process only depends on the quality of X and is independent from its structure. Starting from $\{0\}^n$, if the algorithm carries out such steps at least $b/\delta_{\hat{c}}$ times, it reaches a solution X such that

$$f(X \cup v^*) \geq \left(1 - \left(1 - \alpha_f \frac{b}{bk^*}\right)^{k^*}\right) \cdot f(\tilde{X}_b)$$
$$\geq \left(1 - \frac{1}{e^{\alpha_f}}\right) \cdot f(\tilde{X}_b).$$

Considering item $z = \arg\max_{v \in V: \hat{c}(v) \leq b} f(v)$, by submodularity and $\alpha \in [0,1]$ we have $f(X \cup v^*) \leq (f(X) + f(z))/\alpha_f$.

This implies that

$$\max\{f(X), f(z)\} \geq (\alpha_f/2) \cdot (1 - \frac{1}{e^{\alpha_f}}) \cdot f(\tilde{X}_b).$$

We consider $T = cnP_{\max}B/\delta_{\hat{c}}$ iterations of the algorithm and analyze the success probability within T steps. To have a successful mutation step where v^* is added to the currently best approximate solution, the algorithm has to choose the right individual in the population, which happens with probability at least $1/P_{\max}$. Furthermore, the single bit corresponding to v^* has to be flipped which has the probability at least $1/(en)$. We call such a step a success. Let random variable $Y_j = 1$ when there is a success in iteration j of the algorithm and $Y_j = 0$, otherwise. Thus, we have

$$\Pr(Y_j = 1) \geq \frac{1}{en} \cdot \frac{1}{P_{\max}}$$

as long as a ϕ-approximation for bound b has not been obtained.

Furthermore, let Y_i^*, $1 \leq i \leq T$, be mutually independent random binary variables with $\Pr[Y_i^* = 1] = \frac{1}{enP_{\max}}$ and $\Pr[Y_i^* = 0] = 1 - \frac{1}{enP_{\max}}$. For the expected value of the random variable $Y^* = \sum_{j=1}^{T} Y_j^*$ we have:

$$E[Y^*] = \frac{T}{enP_{\max}} = \frac{cB}{e\delta_{\hat{c}}} \geq \frac{cb}{e\delta_{\hat{c}}}.$$

We use Lemma 1 in (Doerr, Happ, and Klein 2011) for moderately correlated variables which allows the use of the following Chernoff bound

$$\Pr\left(Y < (1-\delta)E[Y^*]\right) \leq \Pr\left(Y^* < (1-\delta)E[Y^*]\right)$$
$$\leq e^{-E[Y^*]\delta^2/2}. \quad (2)$$

Using Equation 2 with $\delta = (1 - \frac{e}{c})$, we bound the probability of not finding a ϕ-approximation of $\tilde{X}_b$ in time $T = cnP_{\max}B/\delta_{\hat{c}}$ by

$$\Pr(Y \leq \frac{b}{\delta_{\hat{c}}}) \leq e^{-\frac{(c-e)^2 B}{2ce\delta_{\hat{c}}}} \leq e^{-\frac{(c/2)^2 B}{2ce\delta_{\hat{c}}}}$$
$$\leq e^{-\frac{cB}{8e\delta_{\hat{c}}}} \leq e^{-\frac{B}{\delta_{\hat{c}}}}.$$

Using the union bound and taking into account that there are at most $B/\delta_{\hat{c}}$ different values for b to consider, the probability that there is a $b \in [0, B]$ for which no ϕ-approximation has been obtained is upper bounded by $\frac{B}{\delta_{\hat{c}}} \cdot e^{-\frac{B}{\delta_{\hat{c}}}}$.

This implies that POMC finds a $(\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$-approximate solution with probability at least $1 - \frac{B}{\delta_{\hat{c}}} \cdot e^{-\frac{B}{\delta_{\hat{c}}}}$ for each $b \in [0, B]$. □

Note that if we have $B/\delta_{\hat{c}} \geq \log n$ then the probability of achieving a ϕ-approximation for every $b \in [0, B]$ is $1 - o(1)$. In order to achieve a probability of $1 - o(1)$ for any possible change, we can run the algorithm for $T' = cnP_{\max} \cdot \max\{\log n, \frac{B}{\delta_{\hat{c}}}\}$, $c \geq 8e + 1$, iterations.

Now we consider the performance of POMC in the dynamic version of the problem. In this version, it is assumed that POMC has achieved a population which includes a ϕ-approximation for all budgets $b \in [0, B]$. Reducing the budget from B to B^* implies that a ϕ-approximation for the newly given budget B^* is already contained in the population.

Consideration must be given to the case where the budget increases. Assume that the budget changes from B to $B^* = B + d$ where $d > 0$. We analyze the time until POMC has updated its population such that it contains for any $b \in [0, B^*]$ a ϕ-approximate solution.

We define

$$I_{\max}(b, b') = \max\{i \in [0, b] \mid \exists X \in P, \hat{c}(X) \leq i$$
$$\wedge f(X) \geq \left(1 - \left(1 - \alpha_f \frac{i}{bk}\right)^{k}\right) \cdot f(\tilde{X}_b)$$
$$\wedge f(X) \geq \left(1 - \left(1 - \alpha_f \frac{i}{b'k'}\right)^{k'}\right) \cdot f(\tilde{X}_{b'})\}$$

for some k and k'. The notion of $I_{\max}(b, b')$ enables us to correlate progress in terms of obtaining a ϕ-approximation for budgets b and b'.

Theorem 6. *Let POMC have a population P such that, for every budget $b \in [0, B]$, there is a ϕ-approximation in P. After changing the budget to $B^* > B$, POMC has computed a ϕ-approximation with probability $\Omega(1)$ within $T = cnP_{\max}\frac{d}{\delta_{\hat{c}}}$ steps for every $b \in [0, B^*]$.*

Proof. Let P denote the current population of POMC in which, for any budget $b \leq B$, there is a $\left(1 - (1 - \frac{\alpha_f}{k})^k\right)$-approximate solution for some k.

Let X be the solution corresponding to $I_{\max}(B, B^*)$. Let $v^* = \arg\max_{v \notin X} \frac{f(X \cup v) - f(X)}{\hat{c}(X \cup v) - \hat{c}(X)}$ be the item with the highest marginal contribution which could be added to X and $X' = X \cup v^*$. According to Lemma 3 and Theorem 2 in (Qian et al. 2017b) and the definition of $I_{\max}(B, B^*)$, we have

$$f(X') \geq$$
$$\left(1 - (1 - \alpha_f \frac{I_{\max} + \hat{c}(X') - \hat{c}(X)}{Bk})^k\right) \cdot f(\tilde{X}_B)$$

and

$$f(X') \geq$$
$$\left(1 - (1 - \alpha_f \frac{I_{\max} + \hat{c}(X') - \hat{c}(X)}{B^*k'})^{k'}\right) \cdot f(\tilde{X}_{B^*}).$$

Table 1: Results for influence maximization with dynamic routing constraints

Changes	GGA		AGGA		$POMC_{1000}$		$POMC_{5000}$		$POMC_{10000}$		$POMC^{WP}_{1000}$		$POMC^{WP}_{5000}$		$POMC^{WP}_{10000}$	
	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st
1-25	85.0349	12.88	81.5734	14.07	66.3992	17.95	77.8569	18.76	86.1057	17.22	86.3846	10.76	86.9270	12.86	85.8794	14.69
26-50	100.7344	22.16	96.1386	23.99	104.9102	15.50	117.6439	16.71	122.5604	15.54	110.4279	11.08	115.6766	14.21	120.8651	14.97
51-75	118.1568	30.82	110.4893	29.50	141.8249	5.64	155.2126	5.08	158.7228	5.20	140.7838	5.02	149.7658	5.49	157.6169	5.54
76-100	127.3422	31.14	115.2978	27.66	149.0259	3.36	159.9100	3.28	162.7353	3.65	148.3012	3.47	155.1943	4.04	163.1958	3.74
101-125	132.3502	29.62	116.9768	25.45	150.3415	3.17	160.1367	2.81	161.2852	2.68	148.5254	2.67	155.1104	3.05	162.3770	2.81
126-150	134.5256	27.69	118.6962	24.19	147.8998	7.36	154.7319	8.77	154.1470	7.43	143.4908	7.96	150.7567	7.82	156.0363	8.12
151-175	135.7651	25.89	119.4982	22.85	147.2478	4.68	153.1417	5.32	151.2966	3.17	143.2959	4.79	149.5447	4.87	153.2526	3.85
176-200	135.5133	24.41	119.1491	22.04	139.5072	8.08	143.6928	9.16	143.9832	8.67	134.7968	8.72	140.5930	8.61	144.4088	8.08

This implies that adding v^* to X violates the budget constraint B, otherwise we would have a greater value for $I_{\max}$.

If $I_{\max} + \hat{c}(X') - \hat{c}(X) \geq B^*$, then, similar to the proof of Theorem 5, we have

$$\max\{f(X), f(z)\} \geq (\alpha_f/2) \cdot \left(1 - \frac{1}{e^{\alpha_f}}\right) \cdot f(\tilde{X}_{B^*}).$$

Otherwise, we have

$$f(X') \geq \left(1 - (1 - \alpha_f \frac{B}{B^* k'})^{k'}\right) \cdot f(\tilde{X}_{B^*}).$$

From this point, the argument in the proof of Theorem 5 holds, i.e., POMC obtains, for each value $b \in [B, B^*]$, a ϕ-approximation after $\frac{d}{\delta_{\hat{c}}}$ successes.

Hence, after $T = cnP_{\max} d/\delta_{\hat{c}}$ iterations, for all $b \in [B, B^*]$ with probability $1 - \frac{d}{\delta_{\hat{c}}} \cdot e^{-\frac{d}{\delta_{\hat{c}}}}$, we have a $\phi = (\alpha_f/2)(1 - \frac{1}{e^{\alpha_f}})$-approximation in the population. □

Note that if the dynamic change is sufficiently large such that $\frac{d}{\delta_{\hat{c}}} \geq \log n$, then the probability of having obtained, for every budget $b \in [0, B^*]$, a ϕ-approximation increases to $1 - o(1)$. A success probability of $1 - o(1)$ can be obtained for this magnitude of changes by giving the algorithm time $T' = cnP_{\max} \max\{\log n, \frac{d}{\delta_{\hat{c}}}\}$, $c \geq 8e + 1$.

A special class of known submodular problems is the maximization of a function with a cardinality constraint. In this case, the constraint value can take on at most $n + 1$ different values and we have $P_{\max} \leq n + 1$. Furthermore, we have $\delta = 1$ which leads to the following two corollaries.

Corollary 7. *Consider the static problem with cardinality constraint bound B. POMC computes, for every budget $b \in [0, B]$, a ϕ-approximation within $T = cn^2 \cdot \max\{B, \log n\}$, $c \geq 8e + 1$, iterations with probability $1 - o(1)$.*

Corollary 8. *Consider the dynamic problem with a cardinality constraint B. Assume that P contains a ϕ-approximation for every $b \in [0, B]$. Then after increasing the budget to B^*, POMC computes, for every budget $b \in [0, B^*]$, a ϕ-approximation in time $T = cn^2 \max\{d, \log n\}$, $c \geq 8e + 1$ and $d = |B^* - B|$, with probability $1 - o(1)$.*

Experimental Investigations

We compare the generalized greedy algorithm (GGA) and adaptive generalized greedy algorithm (AGGA) with the POMC algorithm on the submodular influence maximization problem (Zhang and Vorobeychik 2016; Qian et al. 2017b). We consider dynamic variants of the problems where the constraint bound changes over time.

The Influence Maximization Problem

The influence maximization problem aims to identify a set of most influential users in a social network. Given a directed graph $G = (V, E)$ where each node represents a user. Each edge $(u, v) \in E$ has assigned an edge probability $p_{u,v}((u, v) \in E)$. The probability $p_{u,v}$ corresponds to the strengths of influence from user u to user v. The goal is to find a subset $X \subseteq V$ such that the expected number of activated nodes from X, $IC(X)$, is maximized. Given a cost function c and a budget B the submodular optimization problem is formulated as $\arg\max_{X \subseteq V} E[|IC(X)|]$ s.t. $c(X) \leq B$.

We consider two types of cost functions. The routing constraint takes into account the costs of visiting nodes whereas the cardinality constraint counts the number of chosen nodes. For both cost functions, the constraint is met if the cost is at most B.

For more detailed descriptions of the influence maximization through a social network problem we refer the reader to (Kempe, Kleinberg, and Tardos 2015; Zhang and Vorobeychik 2016; Qian et al. 2017b).

For our dynamic constraint bound changes, we follow the approach taken in (Roostapour, Neumann, and Neumann 2018). We assume that the initial constraint bound is $B = 10$ and stays within the interval $[5, 30]$. We consider a sequence of 200 constraint bounds obtained by randomly increasing or decreasing B by a value of 1. The values of B over time used in our studies are shown in Figure 2. For the experimental investigations of POMC, we consider a parameter τ which determines the number of generations between constraint changes. Furthermore, we consider the option of POMC having a warm-up phase where there are no dynamic changes for the first 10000 iterations. This allows POMC to optimize for an extended period for the initial bound. It should be noted that the number of iterations in the warm-up phase and our choices of τ are relatively small compared to the choice of $10eBn^2$ used in (Qian et al. 2017b) for optimizing the static problem with a given fixed bound B. The results are shown in Tables 1 and 2 and we report for each batch of 25 consecutive constraint bound changes the average solution quality and standard deviation.

Influence Maximization with Dynamic Routing Constraints

We investigate the influence maximization for routing constraints based on simulated networks as done for the static case in (Qian et al. 2017b). We consider a social network

Table 2: Results for influence maximization with dynamic cardinality constraints

Changes	GGA		AGGA		POMC_{1000}		POMC_{5000}		POMC_{10000}		$\text{POMC}^{\text{WP}}_{1000}$		$\text{POMC}^{\text{WP}}_{5000}$		$\text{POMC}^{\text{WP}}_{10000}$	
	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st
1-25	130.9410	14.71	130.6550	14.36	84.8898	24.32	114.8272	23.09	121.1330	19.72	125.2047	10.75	128.6376	13.52	129.3003	15.72
26-50	145.6766	20.70	145.0774	20.11	133.2130	14.69	155.4231	13.98	158.0245	14.34	149.1073	10.62	157.2572	13.28	159.3071	13.40
51-75	160.2780	26.86	159.6331	26.50	164.9157	3.84	184.3274	3.45	187.1952	3.68	171.8898	3.46	187.0476	3.99	187.8508	4.26
76-100	167.9512	26.84	167.3365	26.60	171.5600	1.89	189.4834	2.74	189.6107	2.78	176.1166	1.92	190.3793	3.29	191.5821	3.06
101-125	172.1483	25.45	171.6884	25.35	174.3528	2.11	188.2120	2.32	188.7572	2.46	176.9912	2.47	188.6362	2.63	190.1389	2.35
126-150	174.0582	23.77	173.6528	23.72	174.0404	5.88	183.0188	6.65	183.8033	6.47	175.6150	5.17	183.3861	6.82	184.6115	7.09
151-175	175.1998	22.23	174.8330	22.21	174.5846	4.03	181.3669	4.01	188.4192	3.60	175.6140	3.37	181.7484	3.82	182.9550	4.25
176-200	175.1023	20.94	174.7836	20.92	168.8791	8.05	173.8794	7.28	175.2773	7.23	169.8283	6.73	174.5172	7.19	175.1586	7.39

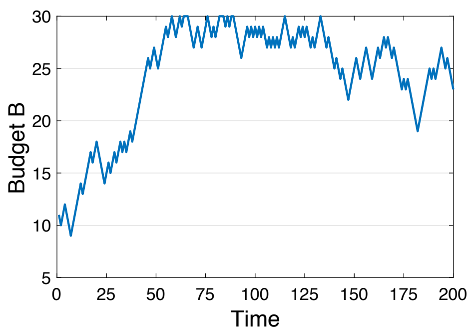

Figure 2: Budget over time for dynamic problems

with 400 nodes that are built using the popular Barabasi-Albert (BA) model (Albert and Barabási 2002) with edge probability $p = 0.1$. The routing network is based on the Erdos-Renyi (ER) model (Erdős and Rényi 1959) where each edge is presented with probability $p = 0.02$. Nodes are placed randomly in the plane and the edge costs are given by Euclidean distances. Furthermore, each chosen node has a cost of 0.1.

We compare the final results for the generalized greedy and adaptive generalized greedy algorithms with our new POMC approach based on the uniform distribution with all the weights being one. Table 1 shows the results of influence spread for the generalized greedy (GGA), adaptive generalized greedy (AGGA), POMC_τ for $\tau = 1000, 5000, 10000$, and $\text{POMC}^{\text{WP}}_\tau$ for $\tau = 1000, 5000, 10000$ and with a warm-up phase (WP) where we run the algorithm for 10000 generation as the initial setting prior to the dynamic process.

Table 1 shows that the generalized greedy algorithm has a better performance than the adaptive one during the dynamic changes. Particularly, after the first 75 changes, the generalized greedy algorithm performs significantly better than the adaptive generalized greedy algorithm. This shows that attempting to adapt to the expected changes results in worse performance than simply starting from scratch. I.e., the cost of adapting outweighs the cost of starting afresh.

Comparing POMC_τ to the greedy approaches we can see that POMC_τ for $\tau = 1000, 5000$ performs worse than the generalized greedy algorithm during the first 25 changes. However, POMC_τ is able to improve its performance over time and outperforms the generalized greedy algorithm after the first 25 changes have occurred. Considering $\text{POMC}^{\text{WP}}_\tau$, it can be observed that the warm-up phase leads to better results than running the generalized greedy algorithm for any interval of changes.

Comparing the different parameter settings used for POMC_τ, it can be observed that POMC_τ for $\tau = 10000$ always performs better than POMC_τ for $\tau = 1000, 5000$ until 125 iteration for dynamic routing constraints. During the changes $126 - 175$, POMC_τ with $\tau = 5000$ achieves better solutions than $\tau = 1000$.

We see that POMC_τ with warm-up for $\tau = 1000, 5000$ outperforms POMC_τ without warm-up within the first 25 iterations. In the other cases, there are no clear differences between running POMC_τ with or without warm-up when considering the same value of τ. This indicates that the algorithm adapts quite quickly to the problem and shows that the warm-up is only necessary for the first 25 changes if the value of τ is quite small.

Influence Maximization with Dynamic Cardinality Constraints

To consider the cardinality constraint, we use the social news data which is collected from the social news aggregator Digg. The Digg dataset contains stories submitted to the platform over a period of a month, and IDs of users who voted on the popular stories. The data consist of two tables that describe friendship links between users and the anonymized user votes on news stories (Hogg and Lerman 2012). As in (Qian et al. 2017b), we use the preprocessed data with 3523 nodes and 90244 edges, and estimated edge probabilities from the user votes based on the method in (Barbieri, Bonchi, and Manco 2012).

The experimental results are shown in Table 2. In contrast to the routing constraint, we see that the generalized greedy algorithm does not have a major advantage over the adaptive generalized greedy algorithm when considering the problem with the cardinality constraint. Although the generalized greedy algorithm achieves better results in all of the intervals, the performance of the adaptive generalized greedy algorithm is highly comparable. In most of the intervals, they achieve almost the same results. The reason for this might be that the change in cost when adding or removing an item is always 1, which makes it less likely for the adaptive generalized greedy algorithm to run into situations where it obtains solutions of low quality. We consider POMC_τ for $\tau = 1000, 5000, 10000$ with and without the warm-up phase. Comparing POMC_τ to the greedy approaches for the problem with the cardinality constraint, we get a similar picture as for the routing constraint. Apart from some exceptions POMC_τ achieves better solutions than the greedy approaches.

POMC$_\tau$ with warm-up phase also clearly outperforms POMC$_\tau$ without warm-up for early changes. Overall, POMC$_\tau$ with warm-up achieves improvements in terms of objective value in most cases over POMC$_\tau$ without warm-up, the generalized greedy approach and the adaptive generalized greedy approach.

Conclusions

Many real-world problems can be modeled as submodular functions and have problems with dynamically changing constraints. We have contributed to the area of submodular optimization with dynamic constraints. Key to the investigations have been the adaptability of algorithms when constraints change. We have shown that an adaptive version of the generalized greedy algorithm frequently considered in submodular constrained optimization is not able to maintain a ϕ-approximation. Furthermore, we have pointed out that the population-based POMC algorithm is able to cater for and recompute ϕ-approximations for related constraints bounds in an efficient way. Our experimental results confirm that advantage of POMC over the considered greedy approaches on important real-world problems. Furthermore, they show that POMC is able to significantly improve its performance over time when dealing with dynamic changes.

Acknowledgement

We thank Chao Qian for providing his POMC implementation and test data to carry out our experimental investigations. This research has been supported by the Australian Research Council (ARC) through grant DP160102401 and the German Science Foundation (DFG) through grant FR2988 (TOSU).

References

Albert, R., and Barabási, A.-L. 2002. Statistical mechanics of complex networks. *Reviews of modern physics* 74(1):47.

Barbieri, N.; Bonchi, F.; and Manco, G. 2012. Topic-aware social influence propagation models. In *IEEE Conference on Data Mining*, 81–90. IEEE Computer Society.

Conforti, M., and Cornuéjols, G. 1984. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete Applied Mathematics* 7(3):251–274.

Doerr, B.; Happ, E.; and Klein, C. 2011. Tight analysis of the (1+1)-EA for the single source shortest path problem. *Evolutionary Computation* 19(4):673–691.

Erdős, P., and Rényi, A. 1959. On random graphs I. *Publicationes Mathematicae Debrecen* 6:290–297.

Friedrich, T., and Neumann, F. 2015. Maximizing submodular functions under matroid constraints by evolutionary algorithms. *Evolutionary Computation* 23(4):543–558.

Friedrich, T.; He, J.; Hebbinghaus, N.; Neumann, F.; and Witt, C. 2010. Approximating covering problems by randomized search heuristics using multi-objective models. *Evolutionary Computation* 18(4):617–633.

Hogg, T., and Lerman, K. 2012. Social dynamics of Digg. *EPJ Data Science* 1(1):5.

Kempe, D.; Kleinberg, J. M.; and Tardos, É. 2015. Maximizing the spread of influence through a social network. *Theory of Computing* 11:105–147.

Krause, A., and Golovin, D. 2014. Submodular function maximization. In Bordeaux, L.; Hamadi, Y.; and Kohli, P., eds., *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press. 71–104.

Laumanns, M.; Thiele, L.; and Zitzler, E. 2004. Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions. *IEEE Trans. Evolutionary Computation* 8(2):170–182.

Lee, J.; Mirrokni, V. S.; Nagarajan, V.; and Sviridenko, M. 2010. Maximizing nonmonotone submodular functions under matroid or knapsack constraints. *SIAM J. Discrete Math.* 23(4):2053–2078.

Lee, J.; Sviridenko, M.; and Vondrák, J. 2010. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.* 35(4):795–806.

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

Qian, C.; Shi, J.; Yu, Y.; Tang, K.; and Zhou, Z. 2017a. Subset selection under noise. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, 3563–3573.

Qian, C.; Shi, J.; Yu, Y.; and Tang, K. 2017b. On subset selection with general cost constraints. In *International Joint Conference on Artificial Intelligence, IJCAI 2017*, 2613–2619.

Qian, C.; Yu, Y.; and Zhou, Z. 2015. Subset selection by Pareto optimization. In Cortes, C.; Lawrence, N. D.; Lee, D. D.; Sugiyama, M.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, 1774–1782.

Roostapour, V.; Neumann, A.; and Neumann, F. 2018. On the performance of baseline evolutionary algorithms on the dynamic knapsack problem. In Auger, A.; Fonseca, C. M.; Lourenço, N.; Machado, P.; Paquete, L.; and Whitley, D., eds., *Parallel Problem Solving from Nature - PPSN XV*, volume 11101 of *LNCS*, 158–169. Springer.

Vondrák, J. 2010. Submodularity and curvature: The optimal algorithm. *RIMS Kôkyûroku Bessatsu* B23:253–266.

Zhang, H., and Vorobeychik, Y. 2016. Submodular optimization with routing constraints. In Schuurmans, D., and Wellman, M. P., eds., *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, 2016*, 819–826. AAAI Press.