Diversifying Greedy Sampling and Evolutionary Diversity Optimisation for Constrained Monotone Submodular Functions

Aneta Neumann
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia

Jakob Bossek
Statistics and Optimization
Dept. of Information Systems
University of Münster
Münster, Germany

Frank Neumann
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia

ABSTRACT

Submodular functions allow to model many real-world optimisation problems. This paper introduces approaches for computing diverse sets of high quality solutions for submodular optimisation problems with uniform and knapsack constraints. We first present diversifying greedy sampling approaches and analyse them with respect to the diversity measured by entropy and the approximation quality of the obtained solutions. Afterwards, we introduce an evolutionary diversity optimisation (EDO) approach to further improve diversity of the set of solutions. We carry out experimental investigations on popular submodular benchmark problems and analyse trade-offs in terms of solution quality and diversity of the resulting solution sets.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**.

KEYWORDS

Evolutionary algorithms, evolutionary diversity optimisation, submodular functions

ACM Reference Format:
Aneta Neumann, Jakob Bossek, and Frank Neumann. 2021. Diversifying Greedy Sampling and Evolutionary Diversity Optimisation for Constrained Monotone Submodular Functions. In *2021 Genetic and Evolutionary Computation Conference (GECCO '21), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3449639.3459385

1 INTRODUCTION

Submodular functions play a key role in the area of optimisation as they allow to model many real-world optimisation problem that encounter diminishing returns [13, 26]. In many real-world applications it is desirable to have a set of high quality solutions available that differ with respect to their design. This allows decision makers to see different options not quantified as part of the formulated objectives and choose from a wide range of options achieving the desired outcome. Computing diverse sets of solutions has gained some attention in the artificial intelligence community with respect to planning problems [8, 9]. In the case of top-k planning, the goal is to obtain a set of solutions such that no better solution outside that set exists [10]. Solutions for this problem are obtained by iteratively forbidding solutions or solution components. Furthermore, the design of diverse sets of policies has been investigated [16] and gradient-based methods have been developed to diversify existing approaches for the creation of policies.

Evolutionary computation approaches provide a more flexible way of creating high quality diverse sets of solutions as diversity measures can directly been imposed on the population. Computing diverse sets of high quality solutions for a given problem has recently gained increasing interest in the area of evolutionary computation under the notions evolutionary diversity optimisation (EDO) [6, 19, 20, 25] and quality diversity (QD) [14, 17]. Both approaches have been mainly used for design problems [7], and only recently been applied to classical combinatorial optimisation problems. In terms of combinatorial optimisation problems, the computation of high quality diverse sets of solutions for the Traveling Salesperson Problem has been considered [2].

In this paper, we propose new approaches for creating diverse sets of high quality solutions for submodular optimisation problems. The classical problem of maximising a monotone submodular function under a given uniform constraint can be solved by a simple greedy algorithm given in [18]. Extensions have been made in terms of knapsack and more general cost constraints [26] as well as results for functions being close to submodular have been obtained. Here generalized greedy approaches that pick in each step an element with a largest marginal gain are used. Furthermore, greedy algorithms also perform well for monotone submodular functions with special types of chance constraints [3].

In recent years, it has been shown for a variety of the beforehand mentioned problems that evolutionary multi-objective algorithms using a Pareto optimisation approach [5, 24] can achieve the same performance guarantees as greedy approaches, but perform much better in practice in a wide range of settings [21, 23]. These approaches relax a given constraint into an additional objective which enables evolutionary multi-objective algorithms to imitate the behaviour of greedy algorithms while also benefiting from local search abilities and interactions in the population. An overview on Pareto optimisation approaches for submodular optimisation can be found in the recent book by Zhou et al. [27].

1.1 Our contribution

We examine approaches for creating diverse sets of high quality solutions for submodular optimisation problems. The class of objective functions that we study are monotone and α_f-submodular

GECCO '21, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8350-9/21/07…$15.00
https://doi.org/10.1145/3449639.3459385

where $\alpha_f \in [0,1]$ measures how close a function f is to being submodular. We are interested in sets of solutions where each solution fulfills a given approximation criterion and aim to maximise the diversity of the set of solutions obtained.

We present approaches for computing diverse sets of solutions for monotone functions under given constraints that all are of high quality. First, we introduce diversifying greedy sampling approaches for monotone functions with a uniform and a knapsack constraint. Our approaches make use of established greedy approaches, but allow to create diverse sets of solutions while still almost maintaining the approximation quality achieved by the greedy approaches. Our theoretical analysis provably shows that solutions constructed by our diversifying greedy sampling approaches only loose a small amount in terms of approximation ratio compared to standard greedy approaches but are able to construct many different solutions.

Afterwards, we introduce an evolutionary diversity optimisation (EDO) approach to maximise diversity under the condition that all solutions fulfill the given quality criterion. We consider the classical entropy measure to measure the diversity of a given set of solutions. Examining the maximal entropy achievable by our greedy sampling approaches, we show that there is room for improving the entropy of the obtained solution sets in many situations. Therefore, we combine the greedy sampling approaches with an EDO approach. Our combined approach for creating high quality diverse sets of solutions first runs one of the developed diversifying greedy sampling approaches and uses EDO to increase the entropy diversity of the obtained set while maintaining the guaranteed approximation quality of the diversifying greedy sampling approaches.

We show in our experimental investigations using real-word graphs that the EDO approach allows to significantly improve the diversity of the set of solutions obtained by diversifying greedy sampling for the classical submodular coverage problem. Furthermore, our experimental results reveal the trade-offs with respect to the diversity of the obtained sets and the quality guaranteed by the diversifying greedy sampling approaches. We also point out how the population size which determines the size of the set of solutions to be obtained increases the diversity of the resulting sets.

The paper is structured as follows. In Section 2, we introduce the problem of computing sets of high quality solutions. In Section 3 and 4, we present our diversifying greedy sampling approaches when working with uniform constraints and knapsack constraints. We examine the short-comings of these approaches and introduce an evolutionary diversity optimisation approach to increase the diversity of our solutions sets in Section 5. In Section 6, we report on our experimental results for different types of submodular optimisation problems. Finally, we finish with some concluding remarks.

2 PROBLEM FORMULATION

Given a set $V = \{v_1, \ldots, v_n\}$, an objective function $f\colon 2^V \to \mathbb{R}_{\geq 0}$, a cost function $c\colon 2^V \to \mathbb{R}_{\geq 0}$ together with a constraint $c(X) \leq B$, the classical goal in optimisation is to find a solution

$$OPT = \arg\max_{X \subseteq V} \{f(X) \mid c(X) \leq B\}.$$

In this paper, we assume that each single element on its own does not violate the constraint, i.e. we have $c(\{v_i\}) \leq B$, $1 \leq i \leq n$, as elements violating the constraint can be completely ignored.

Let A and B be two arbitrary sets such that $A \subseteq B \subseteq V$ holds. The function f is monotone iff we have $f(A) \leq f(B)$. The function f is submodular if for any $v \notin B$,

$$f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B)$$

holds. The submodularity ratio α_f [23, 26] of a given function f is defined as

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

We call a function f, α_f-submodular in this case. Note, that f is submodular iff $\alpha_f = 1$ holds and that we have $\alpha_f \in [0,1]$ in general. Throughout this paper, we assume that f is is monotone and analyze approximation guarantees in terms of α_f.

Given an objective function f and a constraint specified by a cost function c and constraint bound B, we study the problem of computing a (multi-) set of solutions $P = \{P_1, \ldots, P_\mu\}$ such that $f(P_i) \geq f_{min} \wedge c(P_i) \leq B$ holds for $1 \leq i \leq \mu$. Here f_{min} is a threshold value which requires a minimum objective value for any solution in P. For the resulting set P, we are aiming to have a high diversity measured in terms of entropy. We formally introduce our entropy-based diversity measure for a given set of solutions in Section 5.

In the following, we introduce greedy sampling approaches that are able to construct sets of solutions where all solutions have guaranteed quality. We also point out the trade-off in terms of guaranteed quality of the solutions and the number of solutions that are meeting the guaranteed quality. Afterwards, we present a simple evolutionary diversity optimisation approach that can significantly increase the entropy of the solutions sets.

3 DIVERSE SETS FOR UNIFORM CONSTRAINTS

Greedy approaches have been shown to obtain the best possible worst-case performance guarantees for a wide range of constrained monotone submodular problems [18, 26]. In the following, we show how to adapt popular greedy algorithms in the area of submodular optimisation such that they are able to construct diverse sets of high quality solutions. We first consider the case of a uniform constraint, i.e. we have $c(X) = |X| \leq B$.

3.1 Diversifying Greedy Sampling

The diversifying greedy sampling (DGS) approach outlined in Algorithm 1 starts with an empty set and always includes the item with the largest marginal gain until $B - m$ elements have been inserted. The margin m here determines the quality of the partial solution S obtained in this way. After having obtained S using the greedy approach for bound $B - m$, a set of μ solutions $\{P_1, \ldots, P_\mu\}$ is constructed. Each P_i, $1 \leq i \leq \mu$, is constructed by adding m randomly chosen elements from $V \setminus S$ to S.

Throughout this paper, we assume that B and m are positive integers and that m is small compared to B, i.e. $m = o(B)$ holds.

It should be noted that small values of m are already sufficient to construct large sets of diverse solutions if $n - B$ is sufficiently large.

We now analyze DGS with respect to the quality and the diversity of the solution set it obtains. The following theorem shows that the solutions constructed by DGS almost achieve the same worst-case approximation guarantee as the classical greedy approach if m is small compared to B. More precisely, the performance guarantee is only by a factor of $1 - o(1)$ smaller if $m = o(B)$ holds.

Theorem 3.1. *For a given monotone α_f-submodular function f with a uniform constraint $|X| \leq B$ and margin $m \leq B$, DGS computes a population $P = \{P_1, \ldots, P_\mu\}$, where $f(P_i) \geq (1 - e^{-\alpha_f} \cdot (1 - \alpha_f/B)^{-m})) \cdot f(OPT)$, $1 \leq i \leq \mu$. If $m = o(B)$ then for each solution P_i,*

$$f(P_i) \geq \left(1 - e^{-\alpha_f + o(1)}\right) \cdot f(OPT)$$

holds.

Proof. We consider a solution produced by the greedy algorithm which introduced $B - m$ elements. For the proof, we follow the ideas for the classical greedy approach analysed in [18] and take into account the submodularity ratio α_f and the fact that only $B - m$ elements can be introduced to obtain the solution S.

For each set X with $|X| < B - m$ and the element $v^* \notin X$ such that $f(X \cup \{v^*\}) - f(X)$ is maximal, we have

$$\begin{aligned} f(OPT) &\leq f(OPT \cup X) \\ &\leq f(X) + \frac{1}{\alpha_f} \cdot \sum_{v \in (OPT \setminus X)} (f(X \cup \{v\}) - f(X)) \\ &\leq f(X) + (B/\alpha_f) \cdot (f(X \cup \{v^*\}) - f(X)). \end{aligned}$$

Here the first inequality follows from monotonicity, the second one holds as f is α_f-submodular, and the third uses that v^* is an element with the largest marginal gain with respect to f and X and that $|OPT \setminus X| \leq B$ holds.

This implies that in each step an element v^* is added to the current partial solution S, the function value increases by

$$f(S \cup \{v^*\}) - f(S) \geq \frac{\alpha_f}{B} \cdot (f(OPT) - f(S)).$$

Using induction on the number of elements i as done in [18] and taking into account α_f, we obtain a partial solution S_i with

$$f(S_i) \geq (1 - (1 - \alpha_f/B)^i) \cdot f(OPT)$$

After having inserted $B - m$ elements greedily to obtain S, we have

$$\begin{aligned} f(S) &\geq (1 - (1 - \alpha_f/B)^{B-m}) \cdot f(OPT) \\ &= (1 - (1 - \alpha_f/B)^B \cdot (1 - \alpha_f/B)^{-m}) \cdot f(OPT) \\ &\geq (1 - (e^{-\alpha_f} \cdot (1 - \alpha_f/B)^{-m})) \cdot f(OPT). \end{aligned}$$

Assuming $m = o(B)$ and using $\alpha_f \in [0, 1]$, we get

$$\begin{aligned} f(S) &\geq (1 - e^{-\alpha_f} \cdot (1 - 1/B)^{-m})) \cdot f(OPT) \\ &\geq \left(1 - e^{-\alpha_f} \cdot \left((1 - 1/B)^B\right)^{-m/B}\right) \cdot f(OPT) \\ &\geq \left(1 - e^{-\alpha_f} \cdot e^{m/B}\right) \cdot f(OPT) \\ &\geq \left(1 - e^{-\alpha_f + o(1)}\right) \cdot f(OPT). \end{aligned}$$

Each solution P_i is obtained from S by adding m additional elements. As f is monotone, we have $f(P_i) \geq f(S)$ which completes the proof. □

The margin m allows to select m additional elements not contained in S to obtain solutions meeting the quality criteria. This leads to a large number of solutions if the difference $n - B$ is sufficiently large.

Theorem 3.2. *There are at least $\sum_{j=0}^{m} \binom{n-B+m}{j}$ feasible solutions X with $f(X) \geq (1 - e^{-\alpha_f} \cdot (1 - \alpha_f/B)^{-m})) \cdot f(OPT)$. Each P_i produced by DGS is sampled from a set of $\binom{n-B+m}{m}$ such solutions.*

Proof. As $B - m$ elements have only been introduced to obtain S, m additional elements among the $n - B + m$ elements not chosen by the greedy algorithm can be introduced. Adding every possible subset consisting of j, $0 \leq j \leq m$, so far unchosen elements to the $B - m$ elements of S gives a set of

$$\sum_{j=0}^{m} \binom{n - B + m}{j}$$

solutions. Each P_i selects m of such elements which implies that each P_i is sampled from a set of $\binom{n-B+m}{m}$ different solutions. □

Based on the previous proof, we have established that the number of different solutions with exactly B elements that meet the quality criterion of Theorem 3.1 is at least $\binom{n-B+m}{m}$. We will concentrate on such solutions with exactly B elements as we are dealing with monotone functions where the addition of elements does not reduce the function value. In fact, adding elements without violating the constraint often leads to a solution of higher quality in practice.

4 DIVERSE SETS FOR KNAPSACK CONSTRAINTS

We now consider the case of a knapsack constraint. Given a function $\hat{c}\colon V \to \mathbb{R}_{\geq 0}$ assigning non-negative costs to the elements, the cost of a set X is given as $c(X) = \sum_{v \in X} \hat{c}(v)$ and we require $c(X) \leq B$ for a feasible set X.

4.1 Generalized Diversifying Greedy Sampling

The generalized diversifying greedy sampling (GDGS) approach outlined in Algorithm 2 also works with a margin m which reduces the threshold for introducing elements in the greedy steps from B to $B - m$. As in the classical generalized greedy algorithm [26], in each step the element with the largest function gain to cost increase ratio that does not violate the cost constraint is added. The process is iterated until no further element can be added without violating the constraint bound $B - m$ resulting in a set S. The result of the generalized greedy part is a set T which is equal to S or the set with

Algorithm 1: Diversifying Greedy Sampling (DGS)

input: Set of elements V, function f, budget constraint B, margin m, number of solutions μ.

$S \leftarrow \emptyset$;
$V' \leftarrow V$;
while $(|S| + 1 \leq B - m) \wedge (V' \neq \emptyset)$ **do**
  $v^* \leftarrow \arg\max_{v \in V'}(f(S \cup \{v\}) - f(S))$;
  $S \leftarrow S \cup \{v^*\}$;
  $V' \leftarrow V' \setminus \{v^*\}$;
for $i = 1, \ldots, \mu$ **do**
  $P_i \leftarrow S$;
  $V' \leftarrow V \setminus S$;
  while $(|P_i| + 1 \leq B) \wedge (V' \neq \emptyset)$ **do**
    Choose $v^* \in V'$ uniformly at random;
    $P_i \leftarrow P_i \cup \{v^*\}$;
    $V' \leftarrow V' \setminus \{v^*\}$;
return $P = \{P_1, \ldots, P_\mu\}$;

the element y having the largest function value. Among these two options the set with the largest function value is chosen as T. In the sampling part a set of μ solutions $\{P_1, \ldots, P_\mu\}$ is constructed by adding randomly chosen elements to T. Each P_i is obtained by starting with T, choosing an element v^* from the set of remaining elements V' uniformly at random, and adding v^* to P_i if the resulting set does not violate the constraint. The process is iterated until no further elements can be added to P_i.

The following theorem shows that each solution P_i obtained in this way has an approximation guarantee that is only by a minor term smaller than the one of the classical generalized greedy approach working with a budget of B.

Theorem 4.1. *Consider a monotone α_f-submodular function f with a knapsack constraint $c(X) \leq B$ and margin $m \leq B$. If $m = o(B)$, then for each solution P_i, $1 \leq i \leq \mu$, constructed by GDGS, we have*

$$f(P_i) \geq (\alpha_f/2) \cdot \left(1 - e^{-\alpha_f + o(1)}\right) \cdot f(OPT)$$

Proof. We run the generalized greedy algorithm for budget $B - m$. Following [23], the gain in terms of f when inserting v^* is at least

$$f(S \cup \{v^*\}) - f(S) \geq \alpha_f \cdot \frac{c(S \cup \{v^*\}) - c(S)}{B} \cdot (f(OPT) - f(S)).$$

GDGS obtains a solution S with $|S| = L$. Let $z \notin S$ be the first element that has not been added after having produced set S. We have $c(S) + c(\{z\}) > B - m$ and $f(y) \geq f(z)$. Let s_i, $1 \leq i \leq L$, be the ith element added to S and $s_{L+1} = z$. Using induction similar as in [3, 23], we get

Algorithm 2: Generalized Diversifying Greedy Sampling (GDGS)

input: Set of elements V, functions f and c, knapsack constraint B, margin m, number of solutions μ.

$S \leftarrow \emptyset$;
$V' \leftarrow V$;
while $V' \neq \emptyset$ **do**
  $v^* \leftarrow \arg\max_{v \in V'} \frac{f(S \cup \{v\}) - f(S)}{c(S \cup \{v\}) - c(S)}$;
  if $c(S \cup \{v^*\}) \leq B - m$ **then**
    $S \leftarrow S \cup \{v^*\}$;
  $V' \leftarrow V' \setminus \{v^*\}$;
$y \in \arg\max_{v \in V : c(\{v\}) \leq B} f(\{v\})$
$T \leftarrow \arg\max_{Y \in \{S, \{y\}\}} f(Y)$;
for $i = 1, \ldots, \mu$ **do**
  $P_i \leftarrow T$;
  $V' \leftarrow V \setminus T$;
  while $V' \neq \emptyset$ **do**
    Choose $v^* \in V'$ uniformly at random;
    if $c(P_i \cup \{v^*\}) \leq B$ **then**
      $P_i \leftarrow P_i \cup \{v^*\}$;
    $V' \leftarrow V' \setminus \{v^*\}$;
return $P = \{P_1, \ldots, P_\mu\}$;

$$\begin{aligned} f(S \cup \{y\}) &\geq \left[1 - \prod_{j=1}^{L+1} \left(1 - \alpha_f \frac{\hat{c}(s_i)}{B}\right)\right] \cdot f(OPT) \\ &\geq \left[1 - \prod_{j=1}^{L+1} \left(1 - \alpha_f \frac{B - m}{(L+1)B}\right)\right] \cdot f(OPT) \\ &\geq \left[1 - \left(1 - \alpha_f \frac{1}{L+1} + \alpha_f \frac{m}{(L+1) \cdot B}\right)^{L+1}\right] \cdot f(OPT) \end{aligned}$$

Assuming $m = o(B)$ and using $\alpha_f \in [0, 1]$, we have

$$\begin{aligned} f(S \cup \{y\}) &\geq \left[1 - \left(1 - \alpha_f \frac{1}{L+1} + \frac{m}{(L+1) \cdot B}\right)^{L+1}\right] \cdot f(OPT) \\ &\geq \left[1 - \left(1 - \frac{\alpha_f - o(1)}{L+1}\right)^{L+1}\right] \cdot f(OPT) \\ &= \left(1 - e^{-\alpha_f + o(1)}\right) \cdot f(OPT) \end{aligned}$$

Using that f is α_f-submodular, we have

$$\begin{aligned} f(S) + f(\{y\}) &\geq f(S) + \alpha_f \cdot f(\{y\}) \\ &\geq \alpha_f \cdot f(S \cup \{y\}) \\ &\geq \alpha_f \cdot \left(1 - e^{-\alpha_f + o(1)}\right) \cdot f(OPT). \end{aligned}$$

We have $f(T) = \max\{f(S), f(y)\}$ and hence

$$f(T) \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f + o(1)}) \cdot f(OPT).$$

As f is monotone and each P_i is obtained from T by adding an additional set of elements, we have for each i, $1 \leq i \leq \mu$,

$$f(P_i) \geq f(T) \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f + o(1)}) \cdot f(OPT)$$

which completes the proof. □

We now investigate the number of different solutions meeting the quality threshold determined by $B - m$. Let $c_{\min} = \min_{v \in V} \hat{c}(v)$ and $c_{\max} = \max_{v \in V} \hat{c}(v)$ be the minimal and maximal cost of the elements in V, respectively.

Theorem 4.2. *There are at least*

$$\sum_{i=1}^{\lfloor m/c_{\max} \rfloor} \binom{n - \lfloor (B-m)/c_{\min} \rfloor}{i}$$

distinct feasible solutions X with $f(X) \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f + o(1)}) \cdot f(OPT)$.

Proof. Obtaining the set T of costs at most $B - m$, we can pick at least $\lfloor m/c_{\max} \rfloor$ additional elements. For the number of elements in the set T produced by GDGS, we have $|T| \leq \lfloor (B-m)/c_{\min} \rfloor$. This implies that there are at least

$$\sum_{i=1}^{\lfloor m/c_{\max} \rfloor} \binom{n - \lfloor (B-m)/c_{\min} \rfloor}{i}$$

different sets of solutions with the desired approximation quality. □

Note, that in order to obtain μ of these solutions, GDGS creates μ solutions by randomly selecting unchosen elements until no further element can be included without violating the constraint bound B. This implies that in most situations, solutions with a cost close to the constraint bound B are generated.

5 ENTROPY-BASED EVOLUTIONARY DIVERSITY OPTIMISATION

We now introduce a simple EDO approach to create high quality diverse sets of solutions. In order to maximise diversity of our set of μ solutions we introduce a threshold f_{min} and aim to produce a set of solutions P maximising diversity under the condition that for all $x \in P$, $f(x) \geq f_{min}$ holds. The value of f_{min} used in our setting is determined by the introduced greedy sampling approaches and the margin m used and therefore provides a worst-case approximation guarantee for all solutions in the obtained set P.

5.1 Entropy-Based Diversity Measure

Let $V = \{v_1, \ldots, v_n\}$ be the given set of input elements. For a given population P of size μ, we denote by

$$p(v_i) = \frac{|\{P_j \in P \mid v_i \in P_j\}|}{\mu}$$

the fraction of solutions in P that contain element v_i, $1 \leq i \leq n$. We define the entropy of a given population P as

$$H(P) = -\sum_{i=1}^{n} p(v_i) \log_2 p(v_i).$$

Our goal is to maximise the entropy under the condition that all solutions in P are feasible and meet a given quality criterion. We use $\log(x)$ instead of $\log_2(x)$ in the following to simplify notations.

Note, that elements not present in the population or elements present in all solutions have a contribution of 0 to $H(P)$. The function $-x \log x$ is concave in $[0, 1]$ and monotonically increasing in $[0, 1/e]$. This implies that increasing the fraction of solutions in P until a fraction of $1/e$ is obtained is rewarded by the entropy diversity measure. In addition $-x \log x$ is monotonically decreasing in $[1/e, 1]$ which means that increasing fractions to larger than $1/e$ reduces diversity with respect to the considered entropy measure.

5.2 Limitations of Diversifying Greedy Sampling

Before introducing the evolutionary diversity optimisation approach, we point out some limitations of diversifying greedy sampling that show in which way diversity may be improved in common situations while still maintaining high quality solutions. We consider DGS which selects for each solution P_i the same set of elements S with $|S| = B - m$ and adds exactly m additional elements afterwards. This implies that there is no contribution of the elements in S to the entropy score and we can upper bound the entropy of a population as follows. In the following, we assume that μ is large such that each element can be captured in up to a $(1/e)$-fraction of the solutions in the population.

Theorem 5.1. *Let P be a population produced by DGS. Then we have*

$$H(P) \leq -m \log\left(\frac{m}{n - B + m}\right) \leq -\frac{n - B + m}{e} \cdot \log(1/e).$$

Proof. The entropy of a population is maximal if the number of occurrences of all elements in P differs by at most 1 and the fraction of occurrences of all elements is $1/e$. This is due to the fact that the function $f(x) = -x \log x$ is monotonically increasing in $[0, 1/e]$ and monotonically decreasing in $[1/e, 1]$ and that the second derivative $f''(x) = -\frac{1}{\ln(2) \cdot x}$ is negative in $[0, 1]$. W.l.o.g. we assume that the first $B - m$ elements are contained in each of the μ solutions. This means that these elements do not contribute to the entropy measure. The maximum entropy of a population P obtained by DGS is

$$\begin{aligned}
H(P) &= -\sum_{i=1}^{n} p(v_i) \log p(v_i) \\
&= -\sum_{i=B-m+1}^{n} p(v_i) \log p(v_i) \\
&\leq -\sum_{i=B-m+1}^{n} \frac{m}{n - B + m} \log\left(\frac{m}{n - B + m}\right) \\
&= -(n - B + m) \frac{m}{n - B + m} \log\left(\frac{m}{n - B + m}\right) \\
&= -m \cdot \log\left(\frac{m}{n - B + m}\right) \\
&\leq -(n - B + m) \cdot \frac{1}{e} \cdot \log(1/e)
\end{aligned}$$

□

To illustrate the benefit of using the $B - m$ elements of S to improve diversity, we consider the simple example of the function $OneMax(X) = |X|$ which is the simplest non-trivial submodular function.

Assume that B/n is an integer. For OneMax, distributing μB elements equally among the n positions would give a population P^* with $p(v_i) = B/n, 1 \leq i \leq n$, and we have

$$\begin{aligned} H(P^*) &= -\sum_{i=1}^{n} p(v_i) \log p(v_i) \\ &= -\sum_{i=1}^{n} (B/n) \log(B/n) \\ &= -B \log(B/n) \end{aligned}$$

Assuming $B = \lceil n/e \rceil$ (or $B = \lfloor n/e \rfloor$ depending on rounding to the next integer), then $H(P^*)$ is the maximal among all possible populations if each item appears in the population $\lfloor \mu/e \rfloor$ or $\lceil \mu/e \rceil$ times. In this case, the maximal entropy value that can be obtained is approximately $H(P) = -\frac{n}{e} \cdot \log(1/e)$.

5.3 Evolutionary Diversity Optimisation

Our aim is to further improve the sets of solutions created by the diversifying greedy sampling approaches using evolutionary diversity optimisation. Given the set of solutions $P = \{P_1, \ldots, P_\mu\}$ produced by a diversifying greedy sampling approach, we set a quality threshold

$$f_{min} = \min_{i=1,\ldots,\mu} f(P_i)$$

and improve diversity of the set P under the condition that all solutions in P are feasible and have function value at least f_{min}. Note that f_{min} imposes a guaranteed quality dependent on the margin m used in (generalized) diversifying greedy sampling approach as pointed out in Section 3 and 4.

We use the algorithm DIVEA shown in Algorithm 3 to compute a high quality diverse population where each individual I has to meet the given quality criterion f_{min} and does not violate the budget B. An individual $I \subseteq V$ is a set of elements of V. At first, the initial population P is generated with μ individuals created by the diversifying greedy sampling approaches. In each iteration of DIVEA exactly one offspring I' is produced by mutation. We use a mutation operator matching standard bit mutations for bit-strings. Given an individual I, we produce a new individual I' by copying I and changing the status of each $v_i \in V$ for being included or excluded in I' with probably $1/n$. If the offspring I' meets the quality threshold f_{min} and the budget constraint, then I' is added to the population. If I' is added to the population, one individual is selected for removal ensuring that the population size after each iteration is μ. An individual $I \in P$ is removed such that $H(P \setminus \{I\})$ is maximal among all individuals $J \in P$. This implies that the resulting population P has the highest possible entropy of all sets of μ individuals available in iteration t. The algorithm iterates for t_{max} iterations and outputs the final population.

It should be noted that DIVEA is a very simple baseline algorithm and the goal here is to show how evolutionary algorithms can be used to further improve diversity of the sets obtained by the diversifying greedy sampling approaches.

Algorithm 3: Diversifying EA (DIVEA)

input: Initial population $P = \{P_1, \ldots, P_\mu\}$, threshold $f_{min} = \min_{i=1,\ldots,\mu} f(P_i)$, maximum number of iterations t_{max}.

```
t ← 0;
while t ≤ t_max do
    t ← t + 1;
    Choose I ∈ P uniformly at random and produce an
     offspring I' of I by mutation;
    if (f(I') ≥ f_min) ∧ (c(I') ≤ B) then
        P ← P ∪ {I'};
        Remove exactly one individual I, with
         I = arg max_{J∈P} H(P \ {J}) from P;
return P = {P_1, ..., P_μ};
```

6 EXPERIMENTAL INVESTIGATIONS

We examine the introduced algorithms on the submodular influence maximisation problem [15, 23] and the maximum coverage problem [4, 12] with uniform and knapsack constraints. We consider instances of these problems together with a wide range of parameters effecting the diversity entropy score that is achieved. Crucial parameters are the constraint bound B which together with the input size n and the margin m determines the number of guaranteed high quality solutions. The margin m used in the diversifying greedy sampling approach explicitly determines the quality threshold of the diverse set of solutions and the population size μ determines the number of solutions that are contained in the diverse set. As pointed out in Section 3 and 4, an increasing value of m reduces quality but enables higher diversity. Furthermore, a larger value of μ allows to increase the overall entropy diversity score. We discuss the different experimental results for the influence maximisation and maximum coverage problem with a focus on these parameters.

6.1 The Influence Maximisation Problem

The influence maximisation problem (IM) aims to identify a set of the most influential users in a social network. IM intents to maximise the spread of influence through a social network i.e. a graph of social interactions within a group of users [11].

The social network is modeled as a directed graph $G = (V, E)$ where each node represents a user, and each edge $(u, v) \in E$ has been assigned an edge probability $p_{u,v}$ which indicates that user u influences user v. The aim of the influence maximisation problem is to find a subset $X \subseteq V$ such that the number of activated nodes of X is maximised.

6.2 The Maximum Coverage Problem

The maximum coverage problem [4, 12] is a classical NP-hard submodular optimisation problem and arises frequently in a variety of settings. Given a set U of elements, a collection $V = \{V_1, V_2, \ldots, V_n\}$ of subsets of U, a cost function $c : 2^V \to \mathbb{R}_{\geq 0}$ and a budget B, the goal is to find a collection of subsets $X^* \subseteq V$ such that the number of covered elements is maximised subject to meeting the cost

Table 1: Results for the influence maximisation problem with uniform constraints for $B = 10$.

B	m	μ	Threshold (1)		DGS Entropy (2)			DIVEA Entropy (3)		
			mean	std	mean	std	stat	mean	std	stat
10	2	5	43.13	0.1826	4.6039	0.1221	$3^{(-)}$	**23.2193**	1.45E-14	$2^{(+)}$
	2	10	40.07	1.6758	6.5572	0.1456	$3^{(-)}$	**33.1793**	0.2191	$2^{(+)}$
10	2	15	39.46	1.2249	7.7160	0.1353	$3^{(-)}$	**39.0422**	0.1074	$2^{(+)}$
	2	20	39.39	1.3378	8.4572	0.1332	$3^{(-)}$	**43.1293**	0.1826	$2^{(+)}$
10	5	5	40.37	1.8864	11.4096	0.2729	$3^{(-)}$	**23.2193**	1.45E-14	$2^{(+)}$
	5	10	39.22	1.4137	16.2096	0.2828	$3^{(-)}$	**33.2193**	1.45E-14	$2^{(+)}$
10	5	15	38.80	1.2474	18.7255	0.3123	$3^{(-)}$	**39.0645**	0.0243	$2^{(+)}$
	5	20	38.76	1.0637	20.6854	0.2107	$3^{(-)}$	**43.1780**	0.0948	$2^{(+)}$
10	8	5	37.81	1.5747	18.0954	0.5594	$3^{(-)}$	**23.2193**	0.0000	$2^{(+)}$
	8	10	37.52	1.6507	25.2803	0.6395	$3^{(-)}$	**33.2193**	7.49E-15	$2^{(+)}$
10	8	15	37.47	1.5072	29.4682	0.4243	$3^{(-)}$	**39.0689**	7.49E-15	$2^{(+)}$
	8	20	37.27	0.8154	32.2052	0.3433	$3^{(-)}$	**43.2193**	0.0000	$2^{(+)}$

constraint, i.e.

$$X^* = \arg\max_{X \subseteq V} \left\{ \left| \cup_{V_i \in X} V_i \right| \mid c(X) \leq B \right\}.$$

6.3 Experimental Setting

For our experiments, we use the real-world graph frb30-15-01 that contains 450 nodes and 17 827 edges from [22]. In the case of the maximum coverage problem, the set U consists of the vertices of the graph and for each vertex v_i, we construct a set V_i that includes the vertex itself and its adjacent vertices with a higher node number.

We consider uniform and knapsack constraints. In the uniform setting the cost of a set of chosen nodes X for the maximum influence problem is $c(X) = |X|$. In the uniform setting for maximum coverage problem, the cost of a solution X is given by the number of chosen sets, i.e. we have $c(X) = |\{V_i \mid V_i \in X\}|$. In the case of knapsack constraints, the cost of a node v in the influence maximisation problem is given by $\hat{c}(v) = \deg(v) + 1$, where $\deg(v)$ denotes the outdegree of v in G. The cost of a given set of nodes X is given as $c(X) = \sum_{v \in X} \hat{c}(v)$. For the maximum coverage problem, the cost of a set V_i is $\hat{c}(V_i) = |V_i|$ and the cost of a solution X is given as $c(X) = \sum_{V_i \in X} |V_i|$.

6.4 Diverse Sets for Uniform Constraints

We consider the results for the diversifying greedy sampling and diversifying evolutionary algorithm with the uniform constraint. The experimental results of the influence maximisation problem and maximum coverage problem for DGS and DIVEA are shown in Table 1 and Table 2, respectively. For the experimental investigations, we consider all combinations of $m = 2, 5, 8$ and $\mu = 5, 10, 15, 20$ for $B = 10, 15$. For each instance, we run each algorithm 30 times for $10m$ iterations and record the final population.

We compare the results in terms of the entropy values obtained by the DGS and DIVEA at each m and μ for budgets $B = 10, 15$. In order to test the statistical significance of the results we use the Kruskal-Wallis test with 95% confidence in order to measure the statistical validity of our results [1]. $Y^{(+)}$ is equivalent to the statement that the algorithm in the column outperformed algorithm

Table 2: Results for the maximum coverage problem with uniform constraints for $B = 10, 15$.

B	m	μ	Threshold (1)		GDGS Entropy (2)			DIVEA Entropy (3)		
			mean	std	mean	std	stat	mean	std	stat
10	2	5	429.70	1.2360	4.6439	1.80E-15	$3^{(-)}$	**18.0233**	1.9351	$2^{(+)}$
	2	10	428.60	0.9322	6.5972	0.1137	$3^{(-)}$	**23.2874**	1.4800	$2^{(+)}$
10	2	15	427.90	0.9948	7.6921	0.1320	$3^{(-)}$	**25.4377**	1.5628	$2^{(+)}$
	2	20	427.50	1.0009	8.4959	0.9010	$3^{(-)}$	**26.6811**	1.8844	$2^{(+)}$
10	5	5	429.53	1.4320	11.3263	0.2959	$3^{(-)}$	**23.2059**	0.0731	$2^{(+)}$
	5	10	428.83	1.0530	16.0237	0.3192	$3^{(-)}$	**33.1276**	0.1868	$2^{(+)}$
10	5	15	427.80	1.0635	18.7877	0.3068	$3^{(-)}$	**38.6461**	0.3739	$2^{(+)}$
	5	20	427.73	0.8683	20.5862	0.2211	$3^{(-)}$	**42.0368**	0.5832	$2^{(+)}$
10	8	5	383.83	8.2716	18.1037	0.4311	$3^{(-)}$	**23.2193**	1.45E-14	$2^{(+)}$
	8	10	382.77	6.5003	25.2569	0.5274	$3^{(-)}$	**33.2198**	1.45E-14	$2^{(+)}$
10	8	15	378.97	5.4818	29.3765	0.5114	$3^{(-)}$	**39.0689**	2.17E-14	$2^{(+)}$
	8	20	374.90	7.4941	32.1019	0.4154	$3^{(-)}$	**43.2193**	0.000	$2^{(+)}$
15	2	5	449.00	0.0000	4.6305	0.0731	$3^{(-)}$	**27.6151**	2.2429	$2^{(+)}$
	2	10	449.00	0.0000	6.5639	0.5704	$3^{(-)}$	**31.0779**	2.1572	$2^{(+)}$
15	2	15	449.00	0.0000	7.6565	0.1479	$3^{(-)}$	**33.0764**	1.7469	$2^{(+)}$
	2	20	449.00	0.0000	8.5205	0.0917	$3^{(-)}$	**33.7519**	1.8101	$2^{(+)}$
15	5	5	444.17	1.1167	11.3563	0.3401	$3^{(-)}$	**34.6772**	0.3627	$2^{(+)}$
	5	10	443.80	1.0954	16.0621	0.2732	$3^{(-)}$	**47.8240**	1.3957	$2^{(+)}$
15	5	15	443.47	1.0742	18.6366	0.2732	$3^{(-)}$	**53.7846**	1.9886	$2^{(+)}$
	5	20	442.90	0.9595	20.5957	0.2840	$3^{(-)}$	**57.4155**	1.9886	$2^{(+)}$
15	8	5	435.77	1.9061	18.0154	0.4014	$3^{(-)}$	**34.8289**	1.45E-14	$2^{(+)}$
	8	10	434.37	2.2047	25.3553	0.4093	$3^{(-)}$	**49.8223**	0.0365	$2^{(+)}$
15	8	15	434.23	1.9420	29.2837	0.4798	$3^{(-)}$	**58.4817**	0.1664	$2^{(+)}$
	8	20	434.17	1.4162	32.0748	0.4107	$3^{(-)}$	**64.0082**	0.3968	$2^{(+)}$

Y (see numbers behind algorithm names in the top rows of the tables). $Y^{(-)}$ is equivalent to the statement that Y outperformed the algorithm given in the column.

The results in the Table 1 show that the DIVEA for the influence maximisation problem significantly improve the diversity comparing to the DGS approach for $B = 10$. We see a similar result in terms of threshold values where DIVEA is able to attain the quality level achieved by DGA and additionally to produce significantly higher entropy values than DGS for all of the cases. In particular, the DIVEA creates higher entropy values in comparison to the results produced by DSG for the smallest margin $m = 2$. Note that for estimating the influence spread, we simulate the information diffusion process among the users 100 times independently.

Table 2 shows that DIVEA obtains higher entropy values than the DGS among all considered combinations of m and μ. It should be noted that higher entropy values $H(P)$ indicate a higher diversification of the population. In the case of a higher margin m, e.g. $m = 8$, DIVEA improve the diversity consistently. Furthermore, Table 2 includes threshold values obtained by the DGS for all the combinations of m and μ. We observe that the threshold values decrease with increasing size of the margin and number of individuals in the population. The results suggest that the DGS algorithm is able to create diverse solutions and simultaneously maintain a

Table 3: Results for the influence maximisation problem with knapsack constraints for $B = 100$.

B	m	μ	Threshold (1)		GDGS Entropy (2)			DIVEA Entropy (3)		
			mean	std	mean	std	stat	mean	std	stat
100	10	5	48.34	3.0483	2.3422	0.4522	$3^{(-)}$	**14.7342**	2.4239	$2^{(+)}$
	10	10	47.57	3.2168	2.9609	0.4285	$3^{(-)}$	**16.9616**	2.5704	$2^{(+)}$
100	10	15	47.54	2.5498	3.4633	0.5369	$3^{(-)}$	**17.1217**	2.0946	$2^{(+)}$
	10	20	47.50	2.7268	3.4882	0.4566	$3^{(-)}$	**17.7975**	2.2266	$2^{(+)}$
100	20	5	46.04	5.0085	3.2619	0.4084	$3^{(-)}$	**15.7603**	2.4532	$2^{(+)}$
	20	10	45.70	3.3142	4.6998	0.4005	$3^{(-)}$	**18.5986**	2.4380	$2^{(+)}$
100	20	15	45.61	2.1911	5.4380	0.5318	$3^{(-)}$	**19.4041**	1.4929	$2^{(+)}$
	20	20	45.45	2.5654	5.7699	0.4936	$3^{(-)}$	**20.0244**	1.5025	$2^{(+)}$
100	30	5	45.88	1.6549	4.1523	0.5008	$3^{(-)}$	**16.1894**	2.1960	$2^{(+)}$
	30	10	44.03	3.9920	6.1409	0.6590	$3^{(-)}$	**19.4785**	1.8216	$2^{(+)}$
100	30	15	41.62	3.1492	7.0700	0.3484	$3^{(-)}$	**21.4490**	2.0305	$2^{(+)}$
	30	20	40.17	1.7436	7.1638	0.3493	$3^{(-)}$	**22.1779**	1.1061	$2^{(+)}$

Table 4: Results for the maximum coverage problem with knapsack constraints for $B = 100$.

B	m	μ	Threshold (1)		DGS Entropy (2)			DIVEA Entropy (3)		
			mean	std	mean	std	stat	mean	std	stat
100	10	5	406.30	0.7022	2.3190	0.3801	$3^{(-)}$	**5.1566**	1.2727	$2^{(+)}$
	10	10	406.03	0.1826	3.1748	0.2398	$3^{(-)}$	**5.7382**	0.9848	$2^{(+)}$
100	10	15	406.00	0.0000	3.4411	0.2587	$3^{(-)}$	**6.1239**	0.9848	$2^{(+)}$
	10	20	406.00	0.0000	3.6965	0.2307	$3^{(-)}$	**6.6749**	1.0283	$2^{(+)}$
100	20	5	398.53	1.6761	3.5986	0.4343	$3^{(-)}$	**10.0783**	1.4947	$2^{(+)}$
	20	10	397.43	1.0726	5.0012	0.3948	$3^{(-)}$	**11.5237**	1.4219	$2^{(+)}$
100	20	15	397.07	0.9444	5.7040	0.3835	$3^{(-)}$	**11.8965**	0.9952	$2^{(+)}$
	20	20	396.77	0.9714	6.1189	0.4051	$3^{(-)}$	**12.8706**	1.2013	$2^{(+)}$
100	30	5	388.57	2.3294	4.3088	0.6259	$3^{(-)}$	**13.4104**	1.4536	$2^{(+)}$
	30	10	387.93	2.1804	6.2473	0.5190	$3^{(-)}$	**14.7949**	1.3123	$2^{(+)}$
100	30	15	386.97	2.1573	7.2866	0.4094	$3^{(-)}$	**15.7160**	1.0320	$2^{(+)}$
	30	20	386.17	1.7436	7.8370	0.4966	$3^{(-)}$	**16.1779**	1.1061	$2^{(+)}$

similar quality. Predominantly, DIVEA is able to provide the approximation quality achieved by the greedy approaches and to achieve significantly higher entropy values than DGS for all of the cases.

6.5 Diverse Sets for Knapsack Constraints

We now consider the generalized diversifying greedy sampling and diversifying evolutionary algorithm for the setting where the weights depend on the degree of the node of the graph. We consider budgets $B = 100$ and margins $m = 10, 20, 30$. The sizes of the populations is the same as for the uniform constraints. Table 3 and Table 4 include threshold and entropy values obtained by the GDGS and the DIVEA for the combination of m and μ for the influence maximisation problem and maximum coverage problem. For the setting, where B elements can be used, the obtained threshold value for the GDGS and $B = 100$, $\mu = 5$ and $m = 10, 20, 30$ is 48.34, 46.04 and 45.88 for the influence maximisation problem, respectively. Additionally, the threshold value for the GDGS and $B = 100$, $\mu = 5$ and $m = 10, 20, 30$ is 406.30, 398.53 and 388.57 for the maximum coverage problem. Furthermore, we compare the results in terms of the entropy values obtained by the GDGS and the DIVEA. We observe that the DIVEA outperforms the GDGS for all combinations of B, m and μ for both problems. The entropy values of the approaches are overall increasing for each margin value when the number of populations increases. The results also show that the DIVEA is able to more directly improve diversity of the population gathered by the GDGS approach as the number of the margin increases.

7 CONCLUSIONS

We have presented approaches for creating diverse sets of solutions for monotone submodular functions under given constraints. Our diversifying greedy sampling approaches create sets of solutions with provable approximation guarantees that closely match the current best performance ratios obtained by greedy algorithms. Furthermore, we have examined the short-comings in terms of the entropy diversity measure and proposed an entropy-based evolutionary diversity optimisation approach to improve the diversity of the populations obtained by the diversifying greedy sampling approaches. Our experimental results show that high quality sets of solutions can be obtained for important submodular optimisation problems and that the evolutionary diversity optimisation approach significantly increases the entropy diversity of the sets created.

The EDO approach presented in this paper uses a very simple $(\mu + 1)$ EA. An interesting direction for future research is to design complex high performing EDO approaches for submodular optimisation problems based on the introduced setup. Especially, the design of mutation and crossover operators that lead to high quality solutions which are different from the current set of solutions seems to be a crucial component.

8 ACKNOWLEDGEMENTS

This work has been supported by the Australian Research Council through grant DP190103894 and by the South Australian Government through the Research Consortium "Unlocking Complex Resources through Lean Processing".

REFERENCES

[1] Gregory W. Corder and Dale I. Foreman. 2009. *Nonparametric statistics for non-statisticians: a step-by-step approach.* Wiley.
[2] Anh Viet Do, Jakob Bossek, Aneta Neumann, and Frank Neumann. 2020. Evolving diverse sets of tours for the travelling salesperson problem. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2020.* ACM, 681–689. https://doi.org/10.1145/3377930.3389844
[3] Benjamin Doerr, Carola Doerr, Aneta Neumann, Frank Neumann, and Andrew M. Sutton. 2020. Optimization of Chance-Constrained Submodular Functions. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020.* AAAI Press, 1460–1467. https://aaai.org/ojs/index.php/AAAI/article/view/5504
[4] Uriel Feige. 1998. A Threshold of ln *n* for Approximating Set Cover. *J. ACM* 45, 4 (1998), 634–652.
[5] Tobias Friedrich and Frank Neumann. 2015. Maximizing Submodular Functions under Matroid Constraints by Evolutionary Algorithms. *Evolutionary Computation* 23, 4 (2015), 543–558. https://doi.org/10.1162/EVCO_a_00159
[6] Wanru Gao, Samadhi Nallaperuma, and Frank Neumann. 2016. Feature-Based Diversity Optimization for Problem Instance Classification. In *Proceedings of the Parallel Problem Solving from Nature, PPSN 2016 (Lecture Notes in Computer Science)*, Vol. 9921. Springer, 869–879. https://doi.org/10.1007/978-3-319-45823-6_81

[7] Alexander Hagg, Alexander Asteroth, and Thomas Bäck. 2018. Prototype Discovery Using Quality-Diversity. In *Parallel Problem Solving from Nature - PPSN XV - 15th International Conference, 2018, Proceedings, Part I (Lecture Notes in Computer Science)*, Vol. 11101. Springer, 500–511.

[8] Michael Katz and Shirin Sohrabi. 2020. Reshaping Diverse Planning. In *Proceedings of the The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press, 9892–9899.

[9] Michael Katz, Shirin Sohrabi, and Octavian Udrea. 2020. Top-Quality Planning: Finding Practically Useful Sets of Best Plans. In *Proceedings of the The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press, 9900–9907.

[10] Michael Katz, Shirin Sohrabi, Octavian Udrea, and Dominik Winterer. 2018. A Novel Iterative Approach to Top-k Planning. In *Proceedings of the Twenty-Eighth International Conference on Automated Planning and Scheduling, ICAPS 2018*. AAAI Press, 132–140.

[11] David Kempe, Jon M. Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003*. 137–146.

[12] Samir Khuller, Anna Moss, and Joseph Naor. 1999. The Budgeted Maximum Coverage Problem. *Inform. Process. Lett.* 70, 1 (1999), 39–45.

[13] Andreas Krause and Daniel Golovin. 2014. Submodular Function Maximization. In *Tractability: Practical approaches to hard problems*. Cambridge University Press, 71–104.

[14] Joel Lehman and Kenneth O. Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011*. ACM, 211–218.

[15] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne M. VanBriesen, and Natalie S. Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007*. ACM, 420–429.

[16] Muhammad A. Masood and Finale Doshi-Velez. 2019. Diversity-Inducing Policy Gradient: Using Maximum Mean Discrepancy to Find a Set of Diverse Policies. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI 2019*. 5923–5929.

[17] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *CoRR* abs/1504.04909 (2015). http://arxiv.org/abs/1504.04909

[18] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.* 14, 1 (1978), 265–294.

[19] Aneta Neumann, Wanru Gao, Carola Doerr, Frank Neumann, and Markus Wagner. 2018. Discrepancy-based evolutionary diversity optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018*. ACM, 991–998. https://doi.org/10.1145/3205455.3205532

[20] Aneta Neumann, Wanru Gao, Markus Wagner, and Frank Neumann. 2019. Evolutionary diversity optimization using multi-objective indicators. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2019*. ACM, 837–845. https://doi.org/10.1145/3321707.3321796

[21] Aneta Neumann and Frank Neumann. 2020. Optimising Monotone Chance-Constrained Submodular Functions Using Evolutionary Multi-objective Algorithms. In *Proceedings of the Parallel Problem Solving from Nature, PPSN 2020 (Lecture Notes in Computer Science)*, Vol. 12269. Springer, 404–417. https://doi.org/10.1007/978-3-030-58112-1_28

[22] ThanhVu H. Nguyen and Thang Bui. 2018. Benchmark instances. Available at: https://turing.cs.hbg.psu.edu/txn131/.

[23] Chao Qian, Jing-Cheng Shi, Yang Yu, and Ke Tang. 2017. On Subset Selection with General Cost Constraints. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI 2017*. 2613–2619.

[24] Chao Qian, Yang Yu, and Zhi-Hua Zhou. 2015. Subset Selection by Pareto Optimization. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS 2015*. 1774–1782.

[25] Tamara Ulrich and Lothar Thiele. 2011. Maximizing population diversity in single-objective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2011*. ACM, 641–648.

[26] Haifeng Zhang and Yevgeniy Vorobeychik. 2016. Submodular Optimization with Routing Constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI 2016*. AAAI Press, 819–826.

[27] Zhi-Hua Zhou, Yang Yu, and Chao Qian. 2019. *Evolutionary learning: Advances in theories and algorithms*. Springer.