Maximizing Submodular Functions under Matroid Constraints by Evolutionary Algorithms*

Tobias Friedrich
Faculty of Mathematics and Computer Science, Friedrich-Schiller-Universität Jena, 07743 Jena, Germany

Frank Neumann
Optimisation and Logistics, School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia

Abstract

Many combinatorial optimization problems have underlying goal functions that are submodular. The classical goal is to find a good solution for a given submodular function f under a given set of constraints. In this paper, we investigate the runtime of a simple single objective evolutionary algorithm called (1+1) EA and a multi-objective evolutionary algorithm called GSEMO until they have obtained a good approximation for submodular functions. For the case of monotone submodular functions and uniform cardinality constraints we show that GSEMO achieves a $(1-1/e)$-approximation in expected polynomial time. For the case of monotone functions where the constraints are given by the intersection of $k \geq 2$ matroids, we show that the (1+1) EA achieves a $(1/k+\delta)$-approximation in expected polynomial time for any constant $\delta > 0$. Turning to non-monotone symmetric submodular functions with $k \geq 1$ matroid intersection constraints, we show that GSEMO achieves a $1/((k+2)(1+\varepsilon))$-approximation in expected time $\mathcal{O}(n^{k+6}\log(n)/\varepsilon)$.

1 Introduction

Evolutionary algorithms can efficiently find the minima of convex functions. While this is known and well studied in the continuous domain, it is not obvious how an equivalent statement for discrete optimization looks like. Let us recall that a differentiable fitness function $f\colon \mathbb{R} \to \mathbb{R}$ is called *convex* if its derivative $\frac{d}{dx}f(x)$ is non-decreasing in x. The bitstring analogue of this is a fitness function $f\colon \{0,1\}^n \to \mathbb{R}$ whose discrete derivative $\partial_i f(x) = f(x+e_i) - f(x)$ is non-decreasing in x for all $1 \leq i \leq n$ with e_i being the i-th unit vector. A discrete function satisfying the aforementioned condition is called *submodular*. Submodularity is the counterpart of convexity in discrete settings [29].

For understanding the properties of continuous optimizers it is central to study their performance for minimizing convex functions. This has been done in detail for continuous evolutionary algorithms [2, 19]. On the other hand, there is apparently very

*A conference version [11] of this article was presented at PPSN'14.

little prior work on the performance of discrete evolutionary algorithms for optimizing submodular functions. We fill this gap and present several approximation results for simple evolutionary algorithms and submodular functions.

Analogous to the situation for convex functions, there is a significant difference between minimization and maximization of submodular functions. Submodular functions can be *minimized* with a (non-trivial) combinatorial algorithm in polynomial time [22]. On the other hand, submodular function *maximization* is NP-hard as it generalizes many NP-hard combinatorial optimization problems, like maximum cut [10, 17], maximum directed cut [18], maximum facility location [1, 6], and several restricted satisfiability problems [10, 21]. As evolutionary algorithms are especially useful for hard problems, we focus on the maximization of submodular functions.

More formally, we consider the optimization problem $\max\{f(S)\colon S \in \mathcal{I}\}$, where X is an arbitrary ground set, $f\colon 2^X \to \mathbb{R}$ is a fitness function, and $\mathcal{I} \subseteq 2^X$ a collection of independent sets describing the feasible region of the problem. As usual, we assume *value oracle access* to the fitness function; i.e., for a given set S, an algorithm can query an oracle to find its value $f(S)$. We also always assume that the fitness function is normalized, i.e., $f(\emptyset) = 0$, and non-negative, i.e., $f(A) \geq 0$ for all $A \subseteq X$. We will study the following variants of f and $\mathcal{I}$:

- *Submodular functions:* A function f is submodular iff $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for all $A, B \subseteq X$.
- *Monotone functions:* A function f is monotone iff $f(A) \leq f(B)$ for all $A \subseteq B$.
- *Symmetric functions:* A function f is symmetric iff $f(A) = f(X \setminus A)$ for all $A \subseteq X$.
- *Matroid:* A matroid is a pair $(X, \mathcal{I})$ composed of a ground set X and a non-empty collection $\mathcal{I}$ of subsets of X satisfying (1) If $A \in \mathcal{I}$ and $B \subseteq A$ then $B \in \mathcal{I}$ and (2) If $A, B \in \mathcal{I}$ and $|A| > |B|$ then $B + x \in \mathcal{I}$ for some $x \in A \setminus B$. The sets in $\mathcal{I}$ are called *independent*, the *rank* of a matroid is the size of any maximal independent set.
- *Uniform matroid:* A uniform matroid $(X, \mathcal{I})$ of rank $k \in \mathbb{N}$ contains all subsets of size at most k, i.e., $\mathcal{I} = \{A \subseteq X\colon |A| \leq k\}$.
- *Partition matroid:* A partition matroid is a matroid formed from a direct sum of uniform matroids. If the universe X is partitioned into k parts $X_1, \ldots, X_k$ and we have integers d_i with $0 \leq d_i \leq |X_i|$, then in a partition matroid a set I is independent if it contains at most d_i elements from each X_i, i.e., $|I \cap X_i| \leq d_i$ for all i. (Note that in parts of the literature and in the conference version [11] of this paper, it is assumed that $d_i = 1$ for all i.)
- *Intersection of k matroids:* Given k matroids $M_1 = (X, \mathcal{I}_1)$, $M_2 = (X, \mathcal{I}_2)$, …, $M_k = (X, \mathcal{I}_k)$ on the same ground set X, the intersection of these matroids is the matroid $(X, \mathcal{I})$ with $\mathcal{I} = \{A \subseteq X\colon A \in \mathcal{I}_i,\ 1 \leq i \leq k\}$. A simple example for $k = 2$ is the family of matchings in a bipartite graph; or in general the family of hypergraph matchings in a k-partite hypergraph.

Maximizing submodular functions is not only NP-hard, but also NP-hard to approximate. We therefore also have to formalize the notion of an approximation algorithm. We say an algorithm achieves an α-approximation if for all instances of the considered

maximization problem, the output returned by the algorithm is at least α times the optimal value.

Our results. We study the well-known (1+1) EA [8] as well as a multi-objective approach for optimizing submodular functions. Optimizing single objective optimization problems by multi-objective approaches such as the global simple evolutionary multiobjective optimizer (GSEMO) has already been shown to be beneficial for many combinatorial optimization problems [12, 24, 32]. In this article, we prove the following statements.

- Based on the seminal work of Nemhauser, Wolsey, and Fisher [30], we show that GSEMO achieves in polynomial time a $(1 - 1/e)$-approximation for maximizing *monotone submodular* functions under a *uniform matroid constraint* (Theorem 2). This approximation factor is optimal in the general setting [31], and it is optimal even for the special case of Max-r-cover, unless $\mathsf{P} = \mathsf{NP}$ [9]. Furthermore, we show that there are local optima for (1+1) EA which require exponential time to achieve an approximation better than $1/2 + \varepsilon$, for any $\varepsilon > 0$ a constant (Theorem 1).
- Based on recent work of Lee, Sviridenko, and Vondrák [27] and using the idea of p-exchanges, we show that (1+1) EA achieves a $(1/(k+1/p+\varepsilon))$-approximation for any *monotone submodular* function f under k matroid constraints in expected time polynomial in $n^{\mathcal{O}(pk)}$ and $1/\varepsilon$, where $p \geq 1$ is an integer and $\varepsilon > 0$ is a real value (Theorem 3).
- Based on the recent work of Lee, Mirrokni, Nagarajan, and Sviridenko [26], we show that GSEMO achieves in expected time $\mathcal{O}(\frac{1}{\varepsilon} \cdot n^{k+6} \log n)$ a $1/((k+2)(1+\varepsilon))$-approximation for maximizing *symmetric submodular* functions over k *matroid constraints* where $\varepsilon > 0$ is a real value (Theorem 4). Furthermore, we explore the idea of p-exchanges and show that GSEMO obtains (for $k \geq 2$, $p \geq 1$, and $\varepsilon > 0$) a $\left(\frac{1}{((1+\varepsilon)(k+1+1/p)}\right)$-approximation in expected time $\mathcal{O}(\frac{1}{\varepsilon} \cdot n^{2p(k+1)+2} \cdot k \cdot \log n)$ (Theorem 5). Note that these results even hold for *non-monotone* functions.

In the conference version [11] of this article only GSEMO has been studied. This article extends the conference version by providing lower and upper bounds for the (1+1) EA (Section 3.1 and 4) as well as using the idea of p-exchanges to prove improved bounds for GSEMO and the case of symmetric submodular functions in Section 5.

Outline. The paper is organized as follows. In Section 2, we describe the setting for submodular functions and introduce the algorithm that is subject to our investigations. We analyze the algorithm on monotone submodular functions with a uniform constraint in Section 3 and present results for monotone submodular functions under k matroid constraints in Section 4. In Section 5, we consider the case of symmetric (but not necessarily monotane) submodular functions under k matroid constraints. Finally, we finish with a discussion on open problems in Section 6.

2 Preliminaries

Optimization of submodular functions and matroids have received a lot of attention in the classical (non-evolutionary) optimization community. For a detailed exposition, we refer to the textbooks of Korte and Vygen [23] and Schrijver [34].

2.1 Submodular Functions and Matroids

When optimizing a submodular function $f\colon 2^X \to \mathbb{R}$, we will often consider the incremental value of adding a single element. For this, we denote by $F_A(i) = f(A+i) - f(A)$ the marginal value of i with respect to A. Nemhauser et al. [30, Proposition 2.1] give seven equivalent definitions for submodular functions. Additionally to the definition stated in the introduction we will also use that a function f is submodular iff $F_i(A) \geq F_i(B)$ for all $A \subseteq B \subseteq X$ and $i \in X \setminus B$.

Many common pseudo-Boolean and combinatorial fitness functions are submodular. As we are not aware of any general results for the optimization of submodular function by evolutionary algorithms, we list a few examples of well-known submodular functions:

- *Linear functions:* All linear functions $f\colon 2^X \to \mathbb{R}$ with $f(A) = \sum_{i \in A} w_i$ for some weights $w\colon X \to \mathbb{R}$ are submodular. If $w_i \geq 0$ for all $i \in X$, then f is also monotone.

- *Cut:* Given a graph $G = (V, E)$ with nonnegative edge weights $w\colon E \to \mathbb{R}_{\geq 0}$. Let $\delta(S)$ be the set of all edges that contain both a vertex in S and $V \setminus S$. The cut function $w(\delta(S))$ is symmetric and submodular but not monotone.

- *Coverage:* Let the ground set be $X = \{1, 2, \ldots, n\}$. Given a universe U with n subsets $A_i \subseteq U$ for $i \in X$, and a non-negative weight function $w\colon U \to \mathbb{R}_{\geq 0}$. The coverage function $f\colon 2^X \to \mathbb{R}$ with $f(S) = |\bigcup_{i \in S} A_i|$ and the weighted coverage function f' with $f'(S) = w(\bigcup_{i \in S} A_i) = \sum_{u \in \bigcup_{i \in S} A_i} w(u)$ are monotone submodular.

- *Rank of a matroid:* The rank function $r(A) = \max\{|S|\colon S \subseteq A, S \in \mathcal{I}\}$ of a matroid $(X, \mathcal{I})$ is monotone submodular.

- *Hypervolume Indicator:* Given a set of points in $\mathbb{R}^d$ in the objective space of a multi-objective optimization problem, measure the volume of the space dominated by these points relative to some fixed reference point. The hypervolume is a well-known quality measure in evolutionary multi-objective optimization and is known to be monotone submodular [35].

We defined the most important matroids already in the introduction. Matroid theory provides a framework in which many problems from combinatorial optimization can be studied from a unified perspective. Matroids are a special class of so-called *independence systems* that are given by a finite set X and a family of subsets $\mathcal{I} \subseteq X$ such that $\mathcal{I}$ is closed under subsets. Being a matroid is considered to be the property of an independence system which makes greedy algorithms work well. Within evolutionary computation, matroid constraints have been studied only for linear functions [33].

We assume a finite ground set $X = \{x_1, x_2, \ldots, x_n\}$ and identify each subset $S \subseteq X$ with a bitstring $x \in \{0,1\}^n$ such that the i-th bit of x is 1 iff $x_i \in S$. Let $f\colon \{0,1\}^n \to \mathbb{R}_{\geq 0}$ be the given submodular function and $F \subseteq \{0,1\}^n$ be the set of feasible solutions. Note, that f is defined on every element of $\{0,1\}^n$. The constraints determining feasibility are given by k matroids. Given k arbitrary matroids $M_1, \ldots, M_k$ defined on a ground set

Algorithm 1: (1+1) EA Algorithm

```
choose x ∈ {0,1}^n uniformly at random
repeat
    create x' by flipping each bit x_i of x with probability 1/n
    determine h(x')
    if h(x') ≥ h(x) then
        x := x'
until stop
```

X together with their independent systems $I_1, \ldots, I_k$. We consider the problem

$$\max\Big\{f(x)\colon x \in F := \bigcap_{j=1}^{k} I_j\Big\},$$

where f is a submodular function defined on the ground set X.

Intersections of matroids occur in many settings like edge connectivity [14], constrained minimum spanning trees [20] and degree-bounded minimum spanning trees [36].

A prominent example for matroid intersection constraints is the *maximum weight matching problem in bipartite graphs*: Given a bipartite graph $G = (V, E)$ with bipartition $V_1 \cup V_2$, let $M_1 = (X, \mathcal{I}_1)$ and $M_2 = (X, \mathcal{I}_2)$ be two partition matroids on E with

$$\mathcal{I}_1 = \{E' \subseteq E\colon |\delta(v) \cap E'| \leq 1, v \in V_1\},$$
$$\mathcal{I}_2 = \{E' \subseteq E\colon |\delta(v) \cap E'| \leq 1, v \in V_2\},$$

where $\delta(v)$ is the set of neighbors of v. Then it is easy to see that $I \in \mathcal{I}_1 \cap \mathcal{I}_2$ if and only if I induces a matching in G.

Colorful spanning trees are an example for intersecting different kinds of matroids. Let $G = (V, E)$ with edges in E colored with k colors, that is, $E = E_1 \cup E_2 \cup \ldots \cup E_k$. Assume we are given integers $d_1, d_2, \ldots, d_k$ and aim at finding a spanning tree $T \subseteq E$ of G that has at most d_i edges of color i, i.e., $|T \cap E_i| \leq d_i$ for all i. Then this can be phrased as a matroid intersection problem as it is the combination of a spanning tree matroid and a partition matroid.

2.2 Algorithms

The theoretical runtime analysis of evolutionary algorithms often considers randomized local search (RLS) and the $(1+1)$ evolutionary algorithm (EA). We investigate the (1+1) EA (see Algorithm 1) and consider the fitness function $h(x) = (v(x), f(x))$, where $v(x)$ measures the constraint violation of x. Considering problems with k matroid constraints $M_1, \ldots, M_k$, we use

$$v(x) = k \cdot |x|_1 - \sum_{j=1}^{k} r_j(x),$$

where $r_j(x)$ denotes the rank of x in matroid M_j, i.e.

$$r_j(X) = \max\{|Y|\colon Y \subseteq X, Y \in I_j\}$$

Algorithm 2: GSEMO Algorithm

choose $x \in \{0,1\}^n$ uniformly at random
determine $g(x)$
$P \leftarrow \{x\}$
repeat
  choose $x \in P$ uniformly at random
  create x' by flipping each bit x_i of x with probability $1/n$
  determine $g(x')$
  if x' is not strictly dominated by any other search point in P **then**
    include x' into P
    delete all other solutions $z \in P$ with $g(z) \leq g(x')$ from P
until *stop*

for the set X given by x.

We have $v(x) = 0$ iff x is a feasible solution and $v(x) > 0$ otherwise. We optimize $h(x)$ in lexicographic order, i.e.

$$h(y) \geq h(x) \text{ holds iff } (v(y) < v(x)) \vee (v(y) = v(x) \wedge f(y) \geq f(x)).$$

We also consider a multi-objective approach to optimize submodular functions. The multi-objective counterpart of RLS and (1+1) EA are the simple evolutionary multi-objective optimizer (SEMO) [25] and global SEMO (GSEMO) [15]. Both algorithms have been studied in detail, see [5, 7, 12, 15, 16]. We consider the GSEMO given in Algorithm 2. For the multi-objective algorithm, we set $z(x) = f(x)$ iff $x \in F$ and $z(x) = -1$ iff $x \notin F$ and consider the multi-objective problem

$$g(x) := (z(x), |x|_0),$$

where $|x|_0 = \sum_{i=1}^{n}(1 - x_i)$ denotes the number of 0-bits in the given bitstring x. We write $g(x) \geq g(y)$ iff $((z(x) \geq z(y)) \wedge (|x|_0 \geq |y|_0))$ holds. If $g(x) \geq g(y)$ holds, we say that y is dominated by x. The solution y is strictly dominated by solution x iff $g(x) \geq g(y)$ and $g(x) \neq g(y)$. In the end, we focus on the solution $x^* = \arg\max_{x \in P} z(x)$ of GSEMO and study the quality of this solution.

We study the expected number of iterations (of the repeat loop) of (1+1) EA and GSEMO until their feasible solution x^* is for the first time an α-approximation of an optimal feasible solution OPT, i.e. $f(x^*)/\text{OPT} \geq \alpha$ holds. Here α denotes the investigated approximation ratio for the considered problem. We call the expected number of iterations to reach an α-approximation, the expected (run)time to achieve an α-approximation.

3 Monotone Submodular Functions with a Uniform Constraint

In this section, we investigate submodular functions with one uniform constraint. In the case of one uniform constraint of size r, a solution $x \in X$ is feasible if it has at most r elements. Hence, we have $F = \{x \colon x \in X \wedge |x|_1 \leq r\}$.

3.1 Lower bound for (1+1) EA

We consider the (1+1) EA and show that this approach has to cope with local optima with a large inferior neighbourhood. Getting trapped in these local optima, we show that the algorithm finds it hard to achieve an approximation ratio greater than $1/2 + \varepsilon$ where $\varepsilon > 0$ is a constant.

Based on our previously defined fitness function, we have $v(x) = \max\{0, |x|_1 - r\}$ as we are considering problems with one uniform constraint. To show the upper bound on the approximation ratio, we consider an instance of the Max-r-Cover problem.

Our instance is obtained from a bipartite graph which has already been investigated in the context of the vertex cover problem [13]. Let $G = (V_1 \cup V_2, E)$ be the complete bipartite graph on $V_1 = \{v_1, \ldots, v_{\varepsilon n}\}$ and $V_2 = \{v_{\varepsilon n+1}, \ldots, v_n\}$ where $|V_1| = \varepsilon n$ and $|V_2| = (1 - \varepsilon)\, n$ for $\varepsilon < 0.1$. The ground set is given by the set of edges E and each node $v_i \in V_1 \cup V_2$ is identified with the subset of edges adjacent to v_i, i.e. $E_i = \{e \in E \colon e \cap v_i \neq \emptyset\}$. Let E^{V_1} and E^{V_2} be the set of subsets corresponding to the nodes of V_1 and V_2, respectively. We consider the (1+1) EA working with bitstrings of length n where the set E_i is chosen iff $x_i = 1$, $1 \leq i \leq n$.

For the constraint, we set $r = (1-2\varepsilon+\delta)\cdot n$, where δ, $0 < \delta < \varepsilon$, is an arbitrary small positive constant as an upper bound on the number of sets. Furthermore, we require $\varepsilon n \leq r$ which is equivalent to $1-2\varepsilon+\delta \leq \varepsilon$. This implies that E^{V_1} is an optimal solution covering the whole ground set E and can be achieved by setting $1/2 > \varepsilon \geq (1 - \delta)/3$.

We consider the solution x^ℓ where r subsets of E^{V_2} a no subset of E^{V_1} is selected. The value of an optimal solution is $\textsc{Opt} = \varepsilon\,(1-\varepsilon)n^2/2$ and we have $f(x^\ell) = r \cdot (\varepsilon n)/2$. The approximation ratio of x^ℓ is $\alpha(x^\ell) = f(x^\ell)/\textsc{Opt} = (1 - 2\varepsilon + \delta)/(1 - \varepsilon)$. Setting $\varepsilon = (1 - \delta)/3$ we get

$$\begin{aligned} \alpha(x^\ell) &= (1 - 2(1/3 - \delta/3) + \delta)/(1 - 1/3 + \delta/3) \\ &= (1/3 + 5\delta/3)/(2/3 + \delta/3) \\ &= (1 + 5\delta)/(2 + \delta). \end{aligned}$$

Choosing δ as a small constant close to 0, this expression tends to $1/2$.

Theorem 1. *There are monotone submodular functions f for which the (1+1) EA under a uniform matroid constraint may end up in bad local optima. More precisely, there is an instance of the Max-r-Cover problem such that starting with x^ℓ, the expected waiting time for the (1+1) EA to achieve an improvement and therefore a solution with approximation ratio greater than $(1 + 5\delta)/(2 + \delta)$ is $e^{\Omega(n)}$.*

Proof. The search point x^ℓ has r chosen elements and inserting any further elements without removing any other elements is not accepted. Furthermore, removing one or more elements without inserting any new ones covers less elements, which is therefore also not accepted. Each selected set of E^{V_2} covers εn elements which are not covered by any other chosen element whereas each set of E^{V_1} would gain an additional contribution of at most $(1 - \varepsilon - (1 - 2\varepsilon + \delta)) = \varepsilon - \delta$ elements.

In order to have a set of E^{V_1} included and accepted at least δn chosen sets of E^{V_2} have to be removed. Removing δn such elements decreases the fitness by $\delta\varepsilon n^2/2$ and has to be compensated choosing at least δn sets of E^{V_1}.

Hence, in order to have a new accepted solution $2\delta n$ bits have to flip in a mutation step. The probability for this is at most

$$\binom{\varepsilon n}{\delta n} \cdot \binom{(1-\varepsilon)n}{\delta n} \cdot \left(\frac{1}{n}\right)^{2\delta n} = e^{-\Omega(n)}. \qquad \square$$

3.2 Upper Bound for GSEMO

We now turn to GSEMO and show that this approach does not have to cope with local optima that may prevent the algorithm from a achieving an approximation ratio better than $1/2$. GSEMO has the ability of carrying out local search operations, but also allows for a greedy behaviour which is beneficial in this case. The greedy behaviour of GSEMO leads to the following result.

Theorem 2. *The expected time until GSEMO has obtained a $(1 - \frac{1}{e})$-approximation for a monotone submodular function f under a uniform constraint of size r is $\mathcal{O}(n^2 (\log n + r))$.*

Proof. We first study the expected time until GSEMO has produced the solution 0^n for the first time. This solution is Pareto optimal and will therefore stay in the population after it has been produced for the first time. Furthermore, the population size is upper bounded by $n + 1$ as it contains for each i, $0 \leq i \leq n$ at most one solution having exactly i 1-bits. The solution 0^n is feasible and has the maximum number of 0-bits. This implies that the population will not include any infeasible solution to the submodular function f after having included 0^n.

For this step, we consider in each iteration the individual y that has the minimum number of 1-bit among all individuals in the population and denote $\ell = |y|_1$ the number of 1-bits in this individual. Note, that ℓ can not increase during the run of the algorithm. For $1 < \ell \leq n$ a solution y' with $|y'|_1 = \ell - 1$ is produced with probability at least $\ell/(en^2)$ as y' can be produced by selecting y for mutation and flipping one of the ℓ 1-bits. The expected waiting time to include the solution 0^n for the first time into the population is therefore upper bounded by $\sum_{\ell=1}^{n} \left(\frac{\ell}{en^2}\right)^{-1} = \mathcal{O}(n^2 \log n)$.

For the remainder of the proof, we follow the ideas of the proof for the greedy algorithm in Nemhauser et al. [30]. We show that GSEMO produces in expected time $\mathcal{O}(n^2 k)$ for each $0 \leq j \leq r$ a solution X_j with

$$f(X_j) \geq \left(1 - \left(1 - \frac{1}{r}\right)^j\right) \cdot f(\textsc{Opt}), \tag{1}$$

where $f(\textsc{Opt})$ denotes the value of a feasible optimal solution. Note, that a solution is feasible iff it has at most r 1-bits. After having including the solution 0^n into the population this is true for $j = 0$. The proof is done by induction. Assume that GSEMO has already obtained a solution fulfilling Equation (1) for each j, $0 \leq j \leq i < r$. We claim that choosing the solution $x \in P$ with $|x|_1 = i$ for mutation and inserting the element corresponding to the largest possible increase of f increases the value of f by at least $\delta_{i+1} \geq \frac{1}{r} \cdot (f(\textsc{Opt}) - f(X_i))$. Let δ_{i+1} be the increase in f that we obtain when choosing the solution $x \in P$ with $|x|_1 = i$ for mutation and inserting the element corresponding to the largest possible increase.

Due to monotonicity and submodularity, we have $f(\text{OPT}) \leq f(X_i \cup \text{OPT}) \leq f(X_i) + r\delta_{i+1}$ which implies $\delta_{i+1} \geq \frac{1}{r} \cdot (f(\text{OPT}) - f(X_i))$. This leads to

$$f(X_{i+1}) \geq f(X_i) + \frac{1}{r}\left(f(\text{OPT}) - f(X_i)\right) \geq \left(1 - \left(1 - \frac{1}{r}\right)^{i+1}\right) \cdot f(\text{OPT}).$$

For $i = r$, we get $\left(1 - \left(1 - \frac{1}{r}\right)^r\right) \cdot f(\text{OPT}) \geq \left(1 - \frac{1}{e}\right) f(\text{OPT})$. The probability for such a step going from i to $i+1$ is lower bounded by $\frac{1}{en^2}$ and hence the expected time until a $\left(1 - \frac{1}{e}\right)$-approximation has been obtained is at most

$$\mathcal{O}(n^2 \log n) + \sum_{i=0}^{r} \left(\frac{1}{en^2}\right)^{-1} = \mathcal{O}(n^2 (\log n + r)). \qquad \square$$

Max-r-Cover. Let us demonstrate the applicability of Theorem 2 by two examples. First, reconsider the maximum coverage problem introduced in Section 2. Given a universe U with subsets $A_1, A_2, \ldots, A_n \subseteq U$, we want to maximize a coverage function $f(S) = |\bigcup_{i \in S} A_i|$ such that $|S| \leq r$. Theorem 2 immediately implies:

Corollary 1. *The expected time until the GSEMO has obtained a $(1 - 1/e)$-approximation for the Max-r-Cover problem is $\mathcal{O}(n^2 (\log n + r))$. The achieved approximation factor is optimal, unless* $\mathsf{P} = \mathsf{NP}$ *[9].*

Hypervolume indicator. As a second example, we consider a problem from evolutionary multiobjective optimization. As discussed in Section 2, the hypervolume indicator is a monotone submodular function. The hypervolume subset selection problem (HYP-SSP), where we are given n points in $\mathbb{R}^d$ and want to select a subset of size k with maximal hypervolume, therefore aims at maximizing a monotone submodular function $f\colon \{0,1\}^n \to \mathbb{R}_{\geq 0}$ under a uniform matroid constraint of rank r. Theorem 2 implies therefore:

Corollary 2. *The expected time until the GSEMO has obtained a $(1 - 1/e)$-approximation for HYP-SSP is $\mathcal{O}(n^2 (\log n + r))$.*

For dimensions $d > 2$ this is significantly faster than the best known exact algorithm with runtime $\mathcal{O}(n^k)$ [3]. Note that HYP-SSP can be solved in time $\mathcal{O}(n (k + \log n))$ for $d = 2$ [4].

4 Monotone Submodular Functions under Matroid Constraints

The previous section only studied uniform matroid constraints. We now extend this to general matroids and intersection of k matroids and study monotone submodular functions under constraints given by k matroids $M_1, \ldots, M_k$.

We consider the (1+1) EA and start by analyzing the time until the algorithm has obtained a feasible solution x with $f(x) \geq \text{OPT}/n$. This result will later on serve as the basis for the main result of this section.

Lemma 1. *Let f be a monotone submodular function under $k \geq 1$ Matroid constraints and* OPT *be the value of an optimal solution. The expected time until (1+1) EA has obtained a feasible solution with $f(x) \geq \text{OPT}/n$ is $\mathcal{O}(n^{k+1})$.*

Proof. The (1+1) EA starts with the initial solution chosen uniformly at random. We first consider the expected time until the algorithm has obtained for the first time a feasible solution, i.e. a solution x for which $v(x) = 0$ holds. To do this, we generalize Proposition 10 in [33] to the case of the intersection of k matroids. Suppose that x is an infeasible solution with $\ell = v(x)$. During the optimization process ℓ never increases and there are at least ℓ/k distinct elements that can be removed to decrease ℓ. Hence, the probability of decreasing ℓ is at least

$$\frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{\ell}{ekn}$$

and the expected time until a feasible solution has been produced is upper bounded by

$$\sum_{\ell=1}^{kn} \frac{ekn}{\ell} = \mathcal{O}(kn\,(\log k + \log n)).$$

For the remainder of the proof, we work under the assumption that a feasible solution has already been obtained. Let x be an arbitrary feasible solution and x^* be an optimal solution. Furthermore let a be the element in x^* such that $f(\{a\}) \geq \textsc{Opt}/n$. As f is monotone, we have $f(y) \geq \textsc{Opt}/n$ for any feasible solution containing the element a. According to Theorem 2.1 of [26], a feasible solution y containing a can be obtained from any feasible solution x by introducing a and removing at most k elements from x. The expected waiting time of (1+1) EA for such a $(k+1)$-bit flip is $\mathcal{O}(n^{k+1})$. Altogether, the expected time to produce a feasible solution x with $f(x) \geq \textsc{Opt}/n$ is $\mathcal{O}(n^{k+1})$ as $\mathcal{O}(kn\,(\log k + \log n)) = \mathcal{O}(n^{k+1})$ for any $k \geq 1$. □

In the previous section, we have shown that there are local optima for submodular functions with one uniform constraint which only constitute an approximation ratio of most $1/2 + \delta$. Furthermore, the (1+1) EA requires exponential time to leave these local optima. The following theorem shows that the (1+1) EA obtains a $1/(k + \delta)$-approximation for any constant $k \geq 2$ and δ in expected polynomial time. For the case $k = 1$, this implies a $1/(2 + \delta)$-approximation in expected polynomial time as may duplicate the single matroid constraining the search space.

Theorem 3. *For any integers $k \geq 2$, $p \geq 1$ and real value $\varepsilon > 0$, the expected time until the (1+1) EA has obtained a $(1/(k + 1/p + \varepsilon))$-approximation for any monotone submodular function f under k matroid constraints is $\mathcal{O}(\frac{1}{\varepsilon} \cdot n^{2p(k+1)+1} \cdot k \cdot \log n)$.*

Proof. Due to Lemma 1, a feasible solution x with $f(x) \geq \textsc{Opt}/n$ is obtained in expected time $\mathcal{O}(n^{k+1})$. In the following, we work under the assumption that the algorithm has obtained a feasible solution x with $f(x) \geq \textsc{Opt}/n$. A p-exchange operation applied to the current solution x introduces at most $2p$ new elements and deletes at most $2kp$ elements of x. A solution y that can be obtained from x by a p-exchange operation is called a p-exchange neighbour of x. According to [27], every solution x for which there exists no p-exchange neighbour y with $f(y) \geq (1 + \frac{\varepsilon}{n(k+1)}) \cdot f(x)$ is a $(1/(k + 1/p + \varepsilon))$-approximation for any monotone submodular function.

The expected waiting time for a specific p-exchange operation is $\mathcal{O}(n^{2p(k+1)})$ as the probability for a specific p-exchange is $\Omega(n^{-2p(k+1)})$. The number of steps producing

from a solution x a solution y with $f(y) \geq \left(1 + \frac{\varepsilon}{n(k+1)}\right) \cdot f(x)$ is at most

$$\log_{1+\frac{\varepsilon}{n(k+1)}} \frac{\text{OPT}}{\text{OPT}/n} = \mathcal{O}\left(\frac{1}{\varepsilon}\, n\,(k+1) \log n\right).$$

Altogether, the expected time until (1+1) EA has obtained a $(1/(k + 1/p + \varepsilon))$-approximation is $\mathcal{O}(\frac{1}{\varepsilon} \cdot n^{2p(k+1)+1} \cdot k \cdot \log n)$. □

Colorful spanning trees. Recall the example of finding colorful spanning trees from Section 2, which can be describes as a monotone submodular maximization problem under $k = 2$ Matroid constraints. By choosing $p > 1/\varepsilon$ sufficiently large, we get the following corollary.

Corollary 3. *The expected time until the (1+1) EA has obtained a $(1/2 - \varepsilon)$-approximation for colorful spanning trees is $\mathcal{O}(\text{poly}(n)/\varepsilon)$ for all $\varepsilon > 0$.*

5 Symmetric Submodular Functions under Matroid Constraints

We now turn to symmetric submodular functions that are not necessarily monotone. For our analysis, we make use of the following lemma that can be obtained from [26].

Lemma 2. *Let x be a solution such that no solution with fitness at least $\left(1 + \frac{\varepsilon}{n^4}\right) \cdot f(x)$ can be achieved by deleting one element or by inserting one element and deleting at most k elements. Then x is a $\left(\frac{1}{(k+2)(1+\varepsilon)}\right)$-approximation.*

Lemma 2 states that there is always the possibility to achieve a certain progress if no good approximation has been obtained. We use this to show the following results for GSEMO. It should be noted that the corresponding Theorem 2 in the conference version [11] is accidentally missing the symmetry condition.

Theorem 4. *The expected time until the GSEMO has obtained a $\left(\frac{1}{(k+2)(1+\varepsilon)}\right)$-approximation for any symmetric submodular function under k matroid constraints is $\mathcal{O}(\frac{1}{\varepsilon} n^{k+6} \log n)$.*

Proof. Following previous investigations, GSEMO introduces the solution 0^n in the population after an expected number of $\mathcal{O}(n^2 \log n)$ steps. This solution is Pareto optimal and will from that point on stay in the population. Furthermore, 0^n is a feasible solution and has the largest possible number of 0-bits. Hence, from the time 0^n has been included in the population, the population will never include infeasible solutions.

Selecting 0^n for mutation and inserting the element that leads to the largest increase in the f-value produces a solution y with $f(y) \geq \text{OPT}/n$. The reason for this is that the number of elements is limited by n and that f is submodular. Having obtained a solution of fitness at least OPT/n, we focus in each iteration on the individual having the largest f-value in P. Due to the selection mechanism of GSEMO a solution with the maximal f-value will always stay in the population and the value will not decrease during the run of the algorithm.

As long as the algorithm has not obtained a solution of the desired quality, it can produce from its solution x with the highest f-value a feasible offspring y such that $f(y) \geq \left(1 + \frac{\varepsilon}{n^4}\right) \cdot f(x)$. The expected waiting time for this event is $\mathcal{O}(n^{k+2})$ as at most $k + 1$ specific bits of x have to be flipped and using the fact that the population size is at most $n + 1$.

Starting with a solution of quality at least $\textsc{Opt}/n$ the number of such steps in order to achieve an optimal solution is upper bounded by

$$\log_{(1+\frac{\varepsilon}{n^4})} \frac{\textsc{Opt}}{\textsc{Opt}/n} = \mathcal{O}\left(\frac{1}{\varepsilon}\, n^4 \log n\right).$$

Hence, the expected time to achieve a $\left(\frac{1}{(k+2)(1+\varepsilon)}\right)$-approximation is $\mathcal{O}(\frac{1}{\varepsilon} n^{k+6} \log n)$. □

Maximum Cut. As an example, let us consider again the NP-hard Maximum Cut problem, where for a given graph $G = (V, E)$ with n vertices and nonnegative edge weights $w\colon E \to \mathbb{R}_{\geq 0}$, we want to maximize the cut function $\delta(S)$ over all $S \subseteq V$ as defined in Section 2. It is known that the greedy algorithm achieves a 0.5-approximation while the best known algorithms achieve a 0.87856-approximation [17]. Theorem 4 immediately implies the following.

Corollary 4. *The expected time until the GSEMO has obtained a $1/(3\,(1+\varepsilon))$-approximation for the Maximum Cut problem is $\mathcal{O}(\frac{1}{\varepsilon} n^7 \log n)$.*

Note that this result is presumably not tight. We conjecture that a less general analysis can show that GSEMO achieves a $1/2$-approximation.

Using the idea of p-exchanges from Theorem 3, we can improve the approximation result of Theorem 4 with an increasing runtime depending on p.

Theorem 5. *For any integers $k \geq 2$, $p \geq 1$ and real value $\varepsilon > 0$, the expected time until the GSEMO has obtained a $\left(\frac{1}{((1+\varepsilon)(k+1+1/p)}\right)$-approximation for any symmetric submodular function under k matroid constraints is $\mathcal{O}(\frac{1}{\varepsilon} \cdot n^{2p(k+1)+2} \cdot k \cdot \log n)$.*

Proof. GSEMO produces a feasible solution x with $f(x) \geq \textsc{Opt}/n$ in expected time $\mathcal{O}(n^2 \log n)$ (see proof of Theorem 4). After GSEMO has obtained a solution x with $f(x) \geq \textsc{Opt}/n$, we focus on the solution with the largest f-value in the population.

According to Lemma 3.2 of Lee et al. [27] for $k \geq 2$, we have

$$(1+\varepsilon)(k+1/p) \cdot f(S) \geq f(C \cup S) + (k-1+1/p) f(S \cap C) \geq f(S \cup C) + f(S \cap C)$$

if there is no p-exchange neighbour T with $f(T) \geq \left(1+\frac{\varepsilon}{n(k+1)}\right) \cdot f(S)$. As f is symmetric, we have $f(S) = f(X \setminus S)$ and adding $f(X \setminus S)$ to both sides yields

$$(1+\varepsilon)(k+1/p+1) f(S) \geq f(X \setminus S) + f(S \cup C) + f(S \cap C) \geq f(C),$$

which implies $f(S)/f(C) \geq 1/((1+\varepsilon)(k+1+1/p))$. The number of improvements by a factor $\left(1+\frac{\varepsilon}{n(k+1)}\right)$ is upper bounded by

$$\log_{1+\frac{\varepsilon}{n(k+1)}} n = \mathcal{O}\left(\frac{1}{\varepsilon}\, n(k+1) \log n\right).$$

Furthermore, the expected waiting time for such an improvement it $\mathcal{O}(n^{2p(k+1)+1})$ as the population size is upper bound by $n+1$ and a specific p-exchange has probability $\Omega(n^{-2p(k+1)})$ (see proof of Theorem 3). This completes the proof. □

6 Discussion and Open Problems

Maximizing submodular functions under matroid constraints is a very general optimization problem which contains many classical combinatorial optimization problems like maximum cut [10, 17], maximum directed cut [18], maximum facility location [1, 6], and others. We presented several positive and negative results for the approximation behavior of the simple evolutionary algorithms in the framework. To the best of our knowledge, this is the first paper on the analysis of evolutionary algorithms optimizing *submodular functions*. The only result on the performance of evolutionary algorithms under *matroid constraints* is by Reichel and Skutella [33]. They showed that a (1+1) EA achieves in polynomial time a $1/k$-approximation for maximizing a linear function subject to k matroid constraints.

This paper gives a first set of results, but also raises many new questions. We briefly name a few:

- We only study the (1+1) EA and SEMO algorithms, but similar results might be possible for population-based algorithms with appropriate diversity measures.
- Our runtime upper bounds might not be tight. It would be interesting to show matching lower bounds, especially for comparing different algorithms and function classes.
- The proven approximation guarantees hold for very general problem classes. Much tighter results should be possible for specific problems like Maximum Cut.
- Minimizing submodular functions is in general simpler than maximizing submodular functions. However, it is not obvious what this implies for evolutionary algorithms minimizing submodular functions.
- Our proofs strongly rely on the greedy-like behavior of SEMO. It might either be possible (i) to prove a general relationship between SEMO and greedy algorithms or (ii) to give an example where SEMO strictly outperforms a greedy strategy.
- We assume value oracle access to the fitness function f. It might be worth studying the black box complexity of submodular functions in the sense of Lehre and Witt [28].
- We studied submodular fitness functions which are either monotone or symmetric. Future work should also cover submodular functions which are neither monotone nor symmetric.

Acknowledgments

The research leading to these results has received funding from the Australian Research Council (ARC) under grant agreement DP140103400 and from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 618091 (SAGE).

References

[1] A. A. Ageev and M. Sviridenko. An 0.828-approximation algorithm for the uncapacitated facility location problem. *Discrete Applied Mathematics*, 93(2-3):149–156, 1999.

[2] H.-G. Beyer and H.-P. Schwefel. Evolution strategies – a comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[3] K. Bringmann and T. Friedrich. An efficient algorithm for computing hypervolume contributions. *Evolutionary Computation*, 18(3):383–402, 2010.

[4] K. Bringmann, T. Friedrich, and P. Klitzke. Two-dimensional subset selection for hypervolume and epsilon-indicator. In *Annual Conference on Genetic and Evolutionary Computation (GECCO)*, pages 589–596. ACM Press, 2014.

[5] D. Brockhoff, T. Friedrich, N. Hebbinghaus, C. Klein, F. Neumann, and E. Zitzler. On the effects of adding objectives to plateau functions. *IEEE Transactions on Evolutionary Computation*, 13(3):591–603, 2009.

[6] G. Cornuejols, M. Fisher, and G. L. Nemhauser. On the uncapacitated location problem. In *Studies in Integer Programming*, volume 1 of *Annals of Discrete Mathematics*, pages 163 – 177. Elsevier, 1977.

[7] B. Doerr, B. Kodric, and M. Voigt. Lower bounds for the runtime of a global multi-objective evolutionary algorithm. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 432–439, 2013.

[8] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theor. Comput. Sci.*, 276(1-2):51–81, 2002.

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

[10] U. Feige and M. X. Goemans. Approximating the value of two power proof systems, with applications to MAX 2SAT and MAX DICUT. In *3rd Israel Symposium on Theory and Computing Systems (ISTCS)*, pages 182–189, 1995.

[11] T. Friedrich and F. Neumann. Maximizing submodular functions under matroid constraints by multi-objective evolutionary algorithms. In *13th International Conference on Parallel Problem Solving from Nature (PPSN)*, volume 8672 of *Lecture Notes in Computer Science*, pages 922–931. Springer, 2014.

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

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

[14] H. N. Gabow. A matroid approach to finding edge connectivity and packing arborescences. *J. Comput. Syst. Sci.*, 50(2):259–273, 1995.

[15] O. Giel. Expected runtimes of a simple multi-objective evolutionary algorithm. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1918–1925, 2003.

[16] O. Giel and P. K. Lehre. On the effect of populations in evolutionary multi-objective optimisation. *Evolutionary Computation*, 18(3):335–356, 2010.

[17] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[18] E. Halperin and U. Zwick. Combinatorial approximation algorithms for the maximum directed cut problem. In *Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1–7, 2001.

[19] N. Hansen. The CMA evolution strategy: a comparing review. In J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*, pages 75–102. Springer, 2006.

[20] R. Hassin and A. Levin. An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. *SIAM J. Comput.*, 33(2):261–268, 2004.

[21] J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[22] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, July 2001.

[23] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 4th edition, 2007.

[24] S. Kratsch and F. Neumann. Fixed-parameter evolutionary algorithms and the vertex cover problem. *Algorithmica*, 65(4):754–771, 2013.

[25] M. Laumanns, L. Thiele, E. Zitzler, E. Welzl, and K. Deb. Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. In *7th International Conference on Parallel Problem Solving from Nature (PPSN)*, pages 44–53, 2002.

[26] J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Forty-first Annual ACM Symposium on Theory of Computing (STOC)*, pages 323–332, 2009.

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

[28] P. K. Lehre and C. Witt. Black-box search by unbiased variation. *Algorithmica*, 64 (4):623–642, 2012.

[29] L. Lovász. Submodular functions and convexity. In A. Bachem, B. Korte, and M. Grötschel, editors, *Mathematical Programming: The State of the Art*. Springer, 1983.

[30] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14(1):265–294, 1978.

[31] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

[32] F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 5(3):305–319, 2006.

[33] J. Reichel and M. Skutella. Evolutionary algorithms and matroid optimization problems. *Algorithmica*, 57(1):187–206, 2010.

[34] A. Schrijver. *Combinatorial Optimization – Polyhedra and Efficiency*. Springer, 2003.

[35] T. Ulrich and L. Thiele. Bounding the effectiveness of hypervolume-based $(\mu + \lambda)$-archiving algorithms. In *6th International Conference on Learning and Intelligent Optimization (LION)*, pages 235–249, 2012.

[36] R. Zenklusen. Matroidal degree-bounded minimum spanning trees. In *Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1512–1521. SIAM, 2012.