Maximizing Submodular Functions under Matroid Constraints by Multi-objective Evolutionary Algorithms

Tobias Friedrich[1] and Frank Neumann[2]

[1] Friedrich-Schiller-Universität Jena, 07743 Jena, Germany
[2] 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 multi-objective evolutionary algorithm called GSEMO until it has 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 time $\mathcal{O}(n^2(\log n + k))$, where k is the value of the given constraint. For the case of non-monotone submodular functions with k matroid intersection constraints, we show that GSEMO achieves a $1/(k+2+1/k+\varepsilon)$-approximation in expected time $\mathcal{O}(n^{k+5}\log(n)/\varepsilon)$.

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 [25].

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, 17]. On the other hand, it is rather surprising that there appears to be not a single published study regarding the performance of discrete evolutionary algorithms for optimizing submodular functions. We want to 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

T. Bartz-Beielstein et al. (Eds.): PPSN XIII 2014, LNCS 8672, pp. 922–931, 2014.

polynomial time [19]. On the other hand, submodular function *maximization* is NP-hard as it generalizes many NP-hard combinatorial optimization problems, like maximum cut [10, 15], maximum directed cut [16], maximum facility location [1, 7], and several restricted satisfiability problems [10, 18]. 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 is monotone iff $f(A) \leq f(B)$ for all $A \subseteq B$.
- *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, i.e., if the universe X is partitioned into k parts $X_1, \ldots, X_k$, then in a partition matroid a set is independent if it contains at most one element from each part.
- *Intersection of k matroids:* Given k matroids $M_1 = (X, \mathcal{I}_1)$, $M_=(X, \mathcal{I}_2), \ldots$, $M_k = (X, \mathcal{I}_k)$ on the same ground set X, the intersection of these matroids is the matroid $(X, \mathcal{I})$ with $\mathcal{I} = \{A \subseteq X \mid A \in \mathcal{I}_i, 1 \leq i \leq k\}$. 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. 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 [11, 21, 28]. We study GSEMO and prove the following statements.

- Based on the seminal work of Nemhauser, Wolsey, and Fisher [26], we show that GSEMO achieves in polynomial time a $1 - 1/e$-approximation for maximizing *monotone submodular* functions under a *uniform matroid constraint.*

This approximation factor is optimal in the general setting [27], and it is optimal even for the special case of Max-k-cover, unless $\mathsf{P} = \mathsf{NP}$ [9].

- Based on the more recent work of Lee, Mirrokni, Nagarajan, and Sviridenko [23], we show that GSEMO achieves in polynomial time a $1/(k+2+1/k+\varepsilon)$-approximation for maximizing *submodular* functions over *k matroid constraints.* Note that this result even holds for *non-monotone* functions.

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 consider the case of non-monotane submodular functions under matroid constraints in Section 4. Finally, we finish with a discussion on open problems in Section 5.

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 Schrijver [30] and Korte and Vygen [20].

Submodular Functions. 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. [26, 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 submodular but not monotone.
- *Coverage:* Let the ground set be $X = \{1, 2, \dots, n\}$. 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 [31].

Matroids. 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 [29].

Fitness function. 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 (normalized and non-negative) submodular function and $F \subseteq \{0,1\}^n$ be the set of feasible solutions. Note, that f is defined on every element of $\{0,1\}^n$. 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 $g(x) \neq g(y)$.

Algorithms. The theoretical runtime analysis of evolutionary algorithms often considers randomized local search (RLS) and the $(1+1)$ evolutionary algorithm (EA). The multi-objective counterpart of RLS and (1+1) EA are the simple evolutionary multi-objective optimizer (SEMO) [22] and global SEMO (GSEMO) [12]. Both algorithms have been studied in detail, see [6, 8, 11–13]. We consider the GSEMO given in Algorithm 1.

In the end, we focus on the solution $x^* = \arg\max_{x \in P} z(x)$ and study the quality of this solution. We study the expected number of iterations (of the repeat loop) of GSEMO until x^* is an α-approximation of an optimal solution $\textsc{Opt}$, i.e. $f(x^*)/\textsc{Opt} \geq \alpha$ 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 k, a solution $x \in X$ is feasible if it has at most k elements. Hence, we have $F = \{x \mid x \in X \wedge |x|_1 \leq k\}$.

Algorithm 1. 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
```

Theorem 1. *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* k *is* $\mathcal{O}(n^2(\log n + k))$.

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. [26]. We show that GSEMO produces in expected time $\mathcal{O}(n^2 k)$ for each $0 \leq j \leq k$ a solution X_j with

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

where $f(\textsc{Opt})$ denotes the value of a feasible optimal solution. Note, that a solution is feasible iff it has at most k 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, $0 \leq j \leq i < k$. 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 $\delta_{i+1} \geq \frac{1}{k} \cdot (f(\textsc{Opt}) - f(X_i))$. 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 $f(\textsc{Opt}) \leq f(X_i \cup \textsc{Opt}) \leq f(X_i) + k\delta_{i+1}$ which implies $\delta_{i+1} \geq \frac{1}{k} \cdot (f(\textsc{Opt} - f(X_i))$. This leads to $f(X_{i+1}) \geq f(X_i) + \frac{1}{k}\,(f(\textsc{Opt}) - f(X_i)) \geq \left(1 - \left(1 - \frac{1}{k}\right)^{i+1}\right) \cdot f(\textsc{Opt})$.

For $i = k$, we get $\left(1 - \left(1 - \frac{1}{k}\right)^{k}\right) \cdot f(\textsc{Opt}) \geq \left(1 - \frac{1}{e}\right) f(\textsc{Opt})$. 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

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

Max-k-Cover. Let us demonstrate the applicability of Theorem 1 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 $|S| \leq k$. Theorem 1 immediately implies:

Corollary 1. *The expected time until the GSEMO has obtained a* $(1 - 1/e)$*-approximation for the Max-k-Cover problem is* $\mathcal{O}(n^2\,(\log n + k))$*. 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 [4, 5, 14], 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 k. Theorem 1 implies therefore:

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

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, 5].

4 Non-monotone Submodular Functions under Matroid Constraints

We now turn to submodular functions that are not necessarily monotone. The constraints are given by k matroids. Given k arbitrary matroids $M_1, \ldots, M_k$ defined on a ground set X together with their independent systems $I_1, \ldots, I_k$.

We consider the problem $\max\left\{f(x) \mid x \in F := \bigcap_{j=1}^{k} I_j\right\}$, where f is a non-negative submodular function defined on the ground set X. Note that this setting is much more general than the one investigated in the previous section.

For our analysis, we make use of the following lemma in [23].

Lemma 1. *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 k elements and deleting one element. Then x is a $\left(\frac{1}{k+2+\frac{1}{k}+\varepsilon}\right)$-approximation.*

Lemma 1 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.

Theorem 2. *The expected time until the GSEMO has obtained a $\left(\frac{1}{k+2+\frac{1}{k}+\varepsilon}\right)$-approximation for any (non necessarily) non-monotone submodular function under k matroid constraints is $\mathcal{O}(\frac{1}{\varepsilon}n^{k+5}\log n)$.*

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 current solution x 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 $\mathcal{O}(n^{k+1})$ as at most $k+1$ specific bits have to be flipped.

Starting with a solution of quality at least OPT/n the number of such steps in order to achieve an optimal solution is upper bounded by $\log_{1+\frac{\varepsilon}{n^4}} \frac{\text{OPT}}{\text{OPT}/n} = \mathcal{O}\left(\frac{1}{\varepsilon}\, n^4 \log n\right)$. Hence, the expected time to achieve a $\left(\frac{1}{k+2+\frac{1}{k}+\varepsilon}\right)$-approximation is $\mathcal{O}(\frac{1}{\varepsilon}n^{k+5}\log n)$. □

As an example, let us consider again the NP-complete 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 [15]. Theorem 2 immediately implies the following.

Corollary 3. *The expected time until the GSEMO has obtained a* $1/(4+\varepsilon)$*-approximation for the Maximum Cut problem is* $\mathcal{O}(\frac{1}{\varepsilon}n^6 \log n)$*.*

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

5 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, 15], maximum directed cut [16], maximum facility location [1, 7], and others. We presented a number of positive results for the approximation behavior of the GSEMO algorithm 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 [29]. 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 leaves many questions open. We briefly name a few:

- We only study the SEMO algorithm, 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 for SEMO hold for very general problem classes. Much tighter results should be possible for specific problems like Maximum Cut.
- For RLS and (1+1)-EA we conjecture an exponential runtime lower bound to obtain the same approximation ratio for maximizing (monotone) submodular function if the (1 + 1)-EA starts at a random (feasible) solution.
- 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 [24].

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

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

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

[4] Bringmann, K., Friedrich, T., Klitzke, P.: Generic postprocessing via subset selection for hypervolume and epsilon-indicator. In: Bartz-Beielstein, T., et al. (eds.) PPSN XIII 2014. LNCS, vol. 8672, pp. 518–527. Springer, Heidelberg (2014)

[5] Bringmann, K., Friedrich, T., Klitzke, P.: Two-dimensional subset selection for hypervolume and epsilon-indicator. In: Annual Conference on Genetic and Evolutionary Computation (GECCO). ACM Press (2014b)

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

[7] Cornuejols, G., Fisher, M., Nemhauser, G.L.: On the uncapacitated location problem. In: Studies in Integer Programming. Annals of Discrete Mathematics, vol. 1, pp. 163–177. Elsevier (1977)

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

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

[10] Feige, U., Goemans, M.X.: 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), pp. 182–189 (1995)

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

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

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

[14] Glasmachers, T.: Optimized approximation sets of low-dimensional benchmark pareto fronts. In: Bartz-Beielstein, T., et al. (eds.) PPSN XIII 2014. LNCS, vol. 8672, pp. 569–578. Springer, Heidelberg (2014)

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

[16] Halperin, E., Zwick, U.: Combinatorial approximation algorithms for the maximum directed cut problem. In: Twelfth Annual Symposium on Discrete Algorithms (SODA), pp. 1–7 (2001)

[17] Hansen, N.: The CMA evolution strategy: a comparing review. In: Lozano, J., Larranaga, P., Inza, I., Bengoetxea, E. (eds.) Towards a New Evolutionary Computation. Advances in estimation of distribution algorithms, pp. 75–102. Springer (2006)

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

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

[20] Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms, 4th edn. Springer (2007)

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

[22] Laumanns, M., Thiele, L., Zitzler, E., Welzl, E., Deb, K.: Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 44–53. Springer, Heidelberg (2002)

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

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

[25] Lovász, L.: Submodular functions and convexity. In: Bachem, A., Korte, B., Grötschel, M. (eds.) Mathematical Programming: The State of the Art. Springer (1983)

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

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

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

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

[30] Schrijver, A.: Combinatorial Optimization – Polyhedra and Efficiency. Springer (2003)

[31] Ulrich, T., Thiele, L.: Bounding the effectiveness of hypervolume-based $(\mu + \lambda)$-archiving algorithms. In: Hamadi, Y., Schoenauer, M. (eds.) LION 2012. LNCS, vol. 7219, pp. 235–249. Springer, Heidelberg (2012)