Approximating Covering Problems by Randomized Search Heuristics Using Multi-Objective Models*

Tobias Friedrich tobias.friedrich@mpi-inf.mpg.de
Max-Planck-Institut für Informatik, Saarbrücken, Germany

Jun He jqh@aber.ac.uk
Department of Computer Science, University of Wales, Aberystwyth, United Kingdom

Nils Hebbinghaus nils.hebbinghaus@mpi-inf.mpg.de
Max-Planck-Institut für Informatik, Saarbrücken, Germany

Frank Neumann frank.neumann@mpi-inf.mpg.de
Max-Planck-Institut für Informatik, Saarbrücken, Germany

Carsten Witt cfw@imm.dtu.dk
DTU Informatics, Technical University of Denmark, Kgs. Lyngby, Denmark

Abstract

The main aim of randomized search heuristics is to produce good approximations of optimal solutions within a small amount of time. In contrast to numerous experimental results, there are only a few theoretical explorations on this subject. We consider the approximation ability of randomized search heuristics for the class of covering problems and compare single-objective and multi-objective models for such problems. For the VERTEXCOVER problem, we point out situations where the multi-objective model leads to a fast construction of optimal solutions while in the single-objective case, no good approximation can be achieved within the expected polynomial time. Examining the more general SETCOVER problem, we show that optimal solutions can be approximated within a logarithmic factor of the size of the ground set, using the multi-objective approach, while the approximation quality obtainable by the single-objective approach in expected polynomial time may be arbitrarily bad.

1 Introduction

Randomized search heuristics have been shown to be very successful when dealing with problems from combinatorial optimization. The general aim of these heuristics is to produce good approximations of optimal solutions within a small amount of time. In contrast to their success reported in several applications, there are only a few rigorous results on the approximation ability of randomized search heuristics (Witt, 2005). Our aim is to study the following question. Is it possible that a multi-objective model of

*A preliminary version of this article appeared in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO 2007)*, Vol. 1, pp. 797–804, ACM Press, New York, 2007.

a single-objective optimization problem leads to better approximations for NP-hard combinatorial optimization problems?

This question is inspired by a recent work of Neumann and Wegener (2006), where they have shown that minimum spanning trees can be computed more easily in a multi-objective model than in a single-objective one. We follow this interesting new research direction by comparing single- and multi-objective models for an important class of NP-hard combinatorial optimization problems. Our investigations concern covering problems which appear in many important real world applications such as the design of Boolean circuits or the construction of timetables.

Covering problems are from a natural point of view single-objective optimization problems and there is always one single optimal objective value that should be computed and for which a corresponding solution should be produced. In multi-objective optimization, there is usually a trade-off between optimizing different objectives. In this case, one is looking for a set of trade-offs such that improving one objective leads to a disadvantage with respect to at least one other objective. The set of these optimal objective vectors is called the Pareto front. The number of different trade-offs possible determines the maximal population size of the most common multi-objective evolutionary algorithms (EAs) that are typically studied in theoretical papers. The population size of a multi-objective model for a single-objective problem is a crucial point when designing multi-objective models, since an exponential population size may prevent the algorithm from being efficient. Multi-objective models for single-objective problems should include the single-objective problem itself as this is the task which has to be solved. Then the population size may slow down the optimization process compared with the single-objective one. In the case that the population size is polynomially bounded, one can often show (under certain assumptions on the problem structure and the algorithm) that in the worst case the process is slowed down by a polynomial factor. In contrast to this, the multi-objective model directs the search in a better way as shown in Neumann and Wegener (2006). In particular, multi-objective models may make randomized search heuristics behave greedily. Greedy algorithms play an important role in the classical design of algorithms (Cormen et al., 2001). Adding this ability to randomized search heuristics may lead to a significant improvement.

We compare simple randomized search heuristics for single-objective optimization with their multi-objective counterparts by rigorous runtime analyses. In recent years, a lot of progress has been made in analyzing simple evolutionary algorithms with respect to their runtime behavior on artificial pseudo-Boolean functions (Droste et al., 2002; Jansen and Wegener, 2001) as well as some well-known combinatorial optimization problems (Giel and Wegener, 2003; Neumann, 2004; Neumann and Wegener, 2004, 2006; Witt, 2005). Most of these results consider exact optimization while the main aim of general search heuristics is to obtain good approximations of optimal solutions in a small amount of time. Analyzing these algorithms with respect to approximability, we are interested in the worst-case approximation ratio that can be achieved within an expected polynomial number of steps.

As a special case of the more general SetCover problem, we examine the computation of a minimum vertex cover in a given undirected graph. The vertex cover problem has recently been studied extensively in the context of single-objective randomized search heuristics (e.g., Oliveto et al., 2007, 2008). We present a multi-objective model whose set of different trade-offs is always linear in the number of vertices. This seems to be a comfortable situation for multi-objective EAs when dealing with single-objective problems. We do not expect the multi-objective EA to outperform the single-objective

EA in any case, as it has to cope with a larger population size. First, we point out simple situations where this leads to a disadvantage for the multi-objective approach compared with the single-objective one. After that, we present situations for the single-objective case where there is a local optima with a large inferior neighborhood. These local optima can have values that are far from the global optimum. In particular, we present a class of instances where the single-objective model does not lead to an approximation factor better than $n^{1-\delta}$ (where n is the number of subsets), for each δ with $0 < \delta < 1$, within an expected polynomial number of steps while the corresponding multi-objective EAs are even able to compute the Pareto front in a small amount of time. Afterward, we consider the more general SETCOVER problem which is hard to approximate within a multiplicative factor better than $\log m$ (Vazirani, 2001), where m is the size of the ground set. For the single-objective approach, we show that the approximation ratio obtainable in expected polynomial time is unbounded. In contrast to this nonapproximability result for the single-objective approach, we point out that the multi-objective model leads to a factor $\mathcal{O}(\log m)$-approximation for the SETCOVER problem, which is the best we can hope.

The outline of the paper is as follows. In Section 2 we introduce the algorithms that are the subject of our investigations. Section 3 compares the different approaches for the VERTEXCOVER problem. In Section 4 we show that the multi-objective approach leads to a factor $\mathcal{O}(\log m)$-approximation for the SETCOVER problem while the approximation ratio achievable by the single-objective approach is unbounded. Finally, we finish with some conclusions.

2 Algorithms

We consider simple multi-objective evolutionary algorithms and compare them with their single-objective counterparts. The algorithm called SEMO (simple evolutionary multi-objective optimizer) has already been discussed for the optimization of pseudo-Boolean functions (Giel, 2003; Laumanns et al., 2004) and for different kinds of spanning tree problems (Neumann, 2004; Neumann and Wegener, 2006). SEMO starts by choosing a solution uniformly at random from the search space $\{0,1\}^n$. This individual constitutes the initial population P and in each step an individual x is chosen uniformly at random from P to produce an offspring x'. This is done by flipping one random bit of x. The offspring is included in the population if and only if it is not dominated by any other search point of P. In the case of minimizing a multi-objective function $f\colon \{0,1\}^n \to \mathbb{R}^k$, a solution y dominates a solution x if and only if $f(y) \leqslant f(x)$ and $f(y) \neq f(x)$. $f(y) \leqslant f(x)$ holds if and only if $f_i(y) \leqslant f_i(x)$ for all $i \in \{1,\ldots,k\}$. This definition can easily be adjusted to multi-objective problems where the aim is to maximize the value of some objectives.

ALGORITHM 1: *SEMO*

1. *Choose* $x \in \{0,1\}^n$ *uniformly at random.*
2. *Determine* $f(x)$.
3. $P \leftarrow \{x\}$.
4. *Repeat*

- *Choose $x \in P$ uniformly at random.*
- *Create x' by flipping one randomly chosen bit of x.*
- *Determine $f(x')$.*
- *If x' is not dominated by any other search point in P, include x' into P and delete all other solutions $z \in P$ with $f(x') \leqslant f(z)$ from P.*

Choosing a single-objective fitness function which should be optimized for SEMO, the algorithm equals the well-known single-objective randomized search heuristic called randomized local search (RLS). As there is a total order on the search points in the single-objective case, RLS works at each time step with a single solution. We can describe RLS as follows.

ALGORITHM 2: *Randomized local search (RLS)*

1. *Choose $x \in \{0,1\}^n$ uniformly at random.*
2. *Repeat*
   - *Create x' by flipping one randomly chosen bit of x.*
   - *If $f(x') \leqslant f(x)$, set $x := x'$.*

In most cases, evolutionary algorithms have the ability to flip more than one bit in the mutation step. Often the following operator is used, leading to more general algorithms.

ALGORITHM 3: *General mutation operator*

- *Create x' by flipping each bit of x with probability $1/n$.*

The (1+1) EA and the global SEMO are the generalized counterparts of RLS and SEMO, respectively. They differ from Algorithms 1 and 2 above by using the more general mutation operator shown in Algorithm 3. There, each bit of the considered search point is flipped with probability $1/n$. Flipping more than one bit in each step allows the algorithm to leave local optima. Another property of this operator is that the probability of sampling an optimal solution is always positive. This implies that the algorithms (1+1) EA and global SEMO converge to optimal solutions.

Our aim is to analyze the introduced algorithms by a rigorous runtime analysis until they have produced good solutions for covering problems. The measure of interest is the number of constructed solutions until certain goals have been achieved. In the case of single-objective optimization, one is often interested in the expected number of constructed solutions until an optimal one has been obtained for the first time. In the context of multi-objective optimization, the expected optimization time equals the expected number of constructed solutions until the population contains for the first time a solution for each objective vector belonging to the Pareto front. Using multi-objective models for single-objective optimization problems, sometimes one might be only interested in one single solution. In this case, it is enough to bound the number of constructed solutions until a single solution with a certain objective value has been obtained.

Most of our investigations consider the approximation ability of the proposed algorithms. The worst-case approximation ratio of an algorithm A for a given minimization

problem R is defined as $\max_{I \in R} \frac{A(I)}{\mathrm{OPT}(I)}$ where $A(I)$ denotes the value obtained by A when applied to an instance I of R and $\mathrm{OPT}(I)$ denotes the value of an optimal solution for the given instance. We are mainly interested in upper and lower bounds for the number of constructed solutions until a certain approximation ratio has been achieved by the introduced algorithms.

3 The VertexCover Problem

The VertexCover problem is a well-known NP-hard combinatorial optimization problem. Given an undirected graph $G = (V, E)$ where $|V| = n$ and $|E| = m$, the aim is to find a subset $V' \subseteq V$ of minimum cardinality such that for each $e \in E$, $e \cap V' \neq \emptyset$ holds. Many simple approximation algorithms achieve a worst-case approximation ratio of 2 (cf. Cormen et al., 2001). For example, such an approximation can be achieved in polynomial time by computing a maximum matching in the given graph and choosing for each edge of the matching the corresponding two vertices. Considering bipartite graphs, the VertexCover problem can be solved in polynomial time using another correspondence between a maximum matching and a minimum vertex cover given by König's theorem (cf. Diestel, 2005). In this case, the number of edges in a maximum matching equals the number of vertices in a minimum vertex cover.

Consider the algorithms of Section 2 for the VertexCover problem. The search space is $\{0, 1\}^n$ where each bit x_i of a solution x corresponds to a vertex $v_i \in V$. The vertex v_i is chosen in the current solution x if $x_i = 1$; otherwise it is unchosen. We use the fitness function considered by He et al. (2005). Denote by $|x|_1$ and $|x|_0$ the number of ones respectively of zeros in a bitstring x. The fitness of a search point x is given by $f(x) = (u(x), |x|_1)$, where $u(x)$ denotes the number of uncovered edges of the solution x. In the case of RLS and the (1+1) EA, the function should be minimized with respect to the lexicographic order. Hence, the first aim is to minimize the number of uncovered edges such that a vertex cover is obtained. Afterward, the aim is to produce a vertex cover by minimizing the number of ones under the condition that the solution is still a vertex cover. Hence, the lexicographic minimization is equivalent to the minimization of a single-objective fitness function with value $M \cdot u(x) + |x|_1$, where $M > n$ is a sufficiently large penalty. There is a total order on the fitness values, and the vector-valued fitness function is only used for the sake of a unified and simplified presentation. In the case of the multi-objective algorithms SEMO and global SEMO, however, both objectives should be optimized at the same time, such that the objective space is not necessarily totally ordered.

We compare RLS with SEMO and the (1+1) EA with global SEMO by runtime analyses. He et al. (2005) have already examined a single objective EA (EA-I) on the fitness function proposed for the (1+1) EA. Their algorithm works with a larger population size and in addition with a crossover operator. EA-I keeps at each time step the current best solution in the population. Additionally, crossover only happens with probability p_c, which is assumed to be a constant. He et al. have shown that their algorithm finds a vertex cover in a number of $\mathcal{O}(nm)$ generations. We show that the expected time until RLS and the (1+1) EA have produced a vertex cover is $\mathcal{O}(n \log n)$. A similar proof can be found in Glaser (2005). Our results directly transfer to EA-I analyzed in He et al. (2005) and improve the upper bound on the number of generations needed to obtain a feasible solution from $\mathcal{O}(mn)$ to $\mathcal{O}(n \log n)$. In addition, we show that this bound is tight for RLS and the (1+1) EA by presenting a worst case example.

THEOREM 1: *The expected time until RLS and the (1+1) EA have produced a (not necessarily minimum) vertex cover is* $\mathcal{O}(n \log n)$.

PROOF: We prove the theorem for the (1+1) EA using the method of the expected multiplicative weight decrease developed in Neumann and Wegener (2004). As the proof only works with 1-bit flips and all 1-bit flips are equally likely, the result also holds for RLS. Choosing all vertices is certainly a vertex cover and each vertex which has not been chosen before and that is incident to an uncovered edge leads to an improvement with respect to the fitness function. Let k be the number of vertices that are incident to at least one uncovered edge. The number of uncovered edges is reduced from $u(x)$ to 0 by these k accepted 1-bit flips. As the prior aim is to minimize the number of uncovered edges, there are no accepted steps increasing the number of uncovered edges. Nonaccepted 1-bit flips contribute a value of 0 to the reduction of the number of uncovered edges. We consider the expected decrease of $u(x)$ of an arbitrary 1-bit flip. Note that the probability of such steps is at least $1/e$. Choosing a 1-bit flip uniformly at random among all 1-bit flips, the expected number of uncovered edges after this step is at most $(1 - 1/n) \cdot u(x)$ and after t steps this expected value is at most $(1 - 1/n)^t \cdot u(x)$. Choosing $t^* = cn \log n$, where c is an appropriate constant, this value is strictly less than $1/2$. As the number of uncovered edges is an integer, the probability of having obtained a vertex cover after t^* 1-bit flips is at least $1/2$ using Markov's inequality. This implies that the expected number of 1-bit flips to obtain a vertex cover is at most $2t^* = \mathcal{O}(n \log n)$ (geometric series). The result follows as the probability of flipping a single bit in the next mutation step is at least $1/e$ and the expected waiting time for this event is therefore upper bounded by e. □

In the following we show that the given upper bound is best possible. In the case that RLS and the (1+1) EA have to flip $\Theta(n)$ bits to obtain an optimal solution from an initial one, a lower bound of $\Omega(n \log n)$ follows easily using the results of the coupon collectors theorem (Motwani and Raghavan, 1995). For the vertex cover problem, we make this precise by considering the complete graph $C = (V, E)$ on n vertices. Each subset of V containing exactly $n - 1$ vertices is a minimum vertex cover of C.

THEOREM 2: *The expected time until RLS and the (1+1) EA have produced a (minimum) vertex cover of* C *is* $\Theta(n \log n)$.

PROOF: Due to Theorem 1, a vertex cover is produced after an expected number of $\mathcal{O}(n \log n)$ steps. This solution is either a minimum vertex cover (contains exactly $n - 1$ vertices) or a nonoptimal one (containing n vertices). In the second case, exactly one arbitrary bit has to flip. The expected wait time for this event is at most e which shows the upper bound. For the lower bound, we use the following observation. In the initial solution at most $\frac{2}{3} \cdot n$ vertices are chosen with high probability using Chernoff bounds (Motwani and Raghavan, 1995). As at least $n - 1$ vertices are contained in each vertex cover, at least $n/3 - 1$ bits have to flip. The probability of not flipping one of these $n/3 - 1$ bits during $c\,n \log n$ steps, where c is an appropriate constant, is bounded from below by a positive constant using the ideas of the coupon collector's theorem (Droste et al., 2002; Motwani and Raghavan, 1995), which completes the proof. □

Global SEMO has to cope with a larger population size than the (1+1) EA. In particular situations, this can lead to a larger expected optimization time. For the graph C,

the number of vertices for each vertex cover is at least $n-1$ and the (1+1) EA can easily produce such a cover by sequentially adding vertices to the currently best solution. In the case of global SEMO, the set of possible trade-offs might be linear in the number of vertices and this can slow down the time to produce a vertex cover. We show that the expected time for global SEMO to produce a vertex cover of C is significantly larger than the one shown for the (1+1) EA.

THEOREM 3: *The expected time until global SEMO has produced a (minimum) vertex cover of C is $\Theta(n^2 \log n)$.*

PROOF: The population size is $\mathcal{O}(n)$ as there are $n+1$ different values for the number of ones in a search point x. The upper bound follows by considering in each step the solution with the smallest number of uncovered edges in the population and using the ideas also used in the proof of Theorem 1.

It remains to show the lower bound. To this end, we apply the following proof strategy. First, we show that the population size grows to $\Omega(n)$ before the distance to the optimum (in terms of wrongly set bits) becomes small, which means less than $2n^{1/4}$ in our specific setting. From that point, a phase of $\Omega(n^2 \log n)$ steps is considered. We show that the distance to the optimum is with high probability only reduced in the phase if the best individual is chosen. This has probability $\mathcal{O}(1/n)$, and reducing a current distance of r has probability $\mathcal{O}(r/n)$. The bound then follows by estimating a harmonic series.

For the detailed proof, we have to rule out certain events that are considered as a failure and (from the perspective of a lower bound) may lead to premature optimization. The initial search point consists of at most $\frac{2}{3} \cdot n$ vertices with high probability using Chernoff bounds (Motwani and Raghavan, 1995). Let $a_{\max}$ denote the maximal number of vertices of one individual in the current population. We consider the phase where $a_{\max} \in \left[\frac{2}{3}n, \frac{3}{4}n\right]$ and show that after this phase the population size is $\Theta(n)$ with probability at least 1/2. The graph C has the following property for our multi-objective model. Each search point x with $|x|_1 = k$, $0 \leqslant k \leqslant n-1$ is Pareto optimal and its objective vector is $f(x) = ((n-k)(n-k-1)/2, k)$ as the set of uncovered edges consists of all edges between the unchosen vertices. This means that mutating a search point x with $|x|_1 = k$ into a search point x' with $|x|_1 = k'$ leads to an increase of the population size if there is no individual with k' 1-bits present in the population. Let us consider only steps that increase $a_{\max}$. We show that the expected increase of $a_{\max}$ in all such steps in the phase $a_{\max} \in \left[\frac{2}{3}n, \frac{3}{4}n\right]$ is bounded by 2. To this end, we compare steps that increase $a_{\max}$ by i to steps that increase $a_{\max}$ by $i+1$. The latter steps are less likely since one of the remaining (at most $\frac{1}{3} \cdot n$) zeros has to be flipped. The probability for this extra flip is at most $\frac{n}{3}/n = \frac{1}{3}$. Thus, the expected increase of $a_{\max}$ in steps increasing $a_{\max}$ is at most 2 (geometric series). Therefore, the average increase of $a_{\max}$ in the phase $a_{\max} \in [\frac{2}{3}n, \frac{3}{4}n]$ is at most 4 with a probability of at least 1/2. It follows that with a probability of at least 1/2, the population size is $\Theta(n)$ when having obtained for the first time a solution with at least $\frac{3}{4} \cdot n$ vertices. Having produced such a solution for the first time, $a_{\max} \leqslant n - 2n^{1/4}$ holds with probability $1 - e^{-\Omega(n)}$ as the probability of flipping $\Theta(n)$ bits in a single mutation step is $e^{-\Omega(n)}$. Hence, we can assume that there are at least $2n^{1/4}$ zeros left in every element of the current population of size $\Theta(n)$.

Let x be the solution in the population with the largest number of ones. Steps leading to a solution z with $|z|_1 > |x|_1$ are essential to obtain a vertex cover as in each vertex cover of C the number of ones is at least $n-1$. Let $r = |x|_0$ and consider the

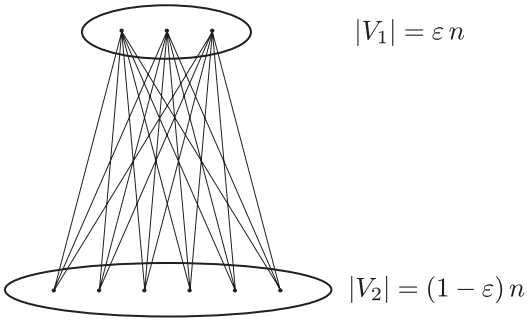

Figure 1: The considered complete bipartite graph $B = (V, E)$ for $n = 9$ and $\varepsilon = \frac{1}{3}$.

time to reduce r from $n^{1/4}$ to 1. The probability to produce from a solution y with $|y|_0 > r + n^{1/4}$ an improving z is $e^{-\Omega(n^{1/4})}$ as the probability of flipping at least $n^{1/4}$ bits in a single mutation step $e^{-\Omega(n^{1/4})}$ and therefore such an event does not happen within a polynomial number of steps with probability exponentially close to 1.

We consider the probability that a mutation flipping at least two 0-bits creates a solution z with $|z|_1 > |x|_1$. The probability to flip at least two 0-bits in a single mutation step of a solution y with $|y|_0 \leqslant r + n^{1/4}$ is upper bounded by $(\frac{r+n^{1/4}}{n})^2 = \mathcal{O}(n^{-6/4})$ and the probability that a solution y with $r \leqslant |y|_0 \leqslant r + n^{1/4}$ is chosen for mutation is $\mathcal{O}(n^{-3/4})$ as the population size is $\Theta(n)$. Hence, a solution z with $|z|_1 > |x|_1$ is produced by flipping at least two 0-bits with probability $\mathcal{O}(n^{-9/4})$ in the next step and it does not happen within a phase of $\Theta(n^2 \log n)$ steps with probability $1 - o(1)$ using Markov's inequality. This implies that with probability $1 - o(1)$, a solution z with $|z|_1 > |x|_1$ can only be produced by mutating one bit of x, where x is still the solution with the largest number of ones. The expected time to reduce the value r of zeros to $r - 1$ zeros under the condition that x has been chosen for mutation is at least $\frac{n}{r}$. Thus, the expected time to reduce the value r from $n^{1/4}$ to 1 is

$$\Omega\left(n \sum_{r=2}^{n^{1/4}} \frac{n}{r}\right) = \Omega(n^2 \log n).$$

This proves an expected time of $\Omega(n^2 \log n)$ to find a vertex cover of C. □

We have shown that there are cases where the population size of global SEMO slows down the optimization process. This seems to be a typical situation for dense graphs that have δn^2, $\delta > 1/4$, edges. In this case, the initial solution of both algorithms does not represent a vertex cover with high probability, and $\Theta(n)$ vertices have to be chosen to obtain such a solution.

In the following, we want to point out a situation where the multi-objective approach is superior. Consider a complete bipartite graph $B = (V, E)$, where $V = V_1 \cup V_2$ consists of two sets of nonequal size and the edge set $E = \{\{v_i, v_j\} \mid v_i \in V_1 \wedge v_j \in V_2\}$ consists of all edges that connect these two sets. W. l. o. g. we assume $|V_1| < |V_2|$. A minimum vertex cover is the set V_1 but both algorithms have a chance to determine the set V_2 as vertex cover. We consider the case $V_1 = \{v_1, v_2, \ldots, v_{\varepsilon n}\}$ and $V_2 = \{v_{\varepsilon n+1}, v_{\varepsilon n+2}, \ldots, v_n\}$,

$0 < \varepsilon < 1/2$ and where ε is not necessarily constant. The usual aim of randomized search heuristics is to produce near optimal solutions. In the following, we point out that the single-objective approach does not admit a good approximation of an optimal solution for the graph B while the multi-objective one leads to a polynomial expected optimization time. If RLS has chosen all vertices of V_2 but some vertices of V_1 are missing, the algorithm cannot produce an approximation better than a factor $\frac{(1-\varepsilon)}{\varepsilon}$.

On the graph, B, the expected optimization time of RLS, is infinite as the next theorem shows.

THEOREM 4: *With probability ε, RLS cannot obtain an approximation better than a factor $(1-\varepsilon)/\varepsilon$ for B within a finite number of steps. In particular, the expected time to produce an approximation better than a factor of $(1-\varepsilon)/\varepsilon$ on B is infinite.*

For the proof of Theorem 4, we will use the following lemma which may be of independent interest.

LEMMA 1: *A bin contains k red and ℓ blue balls. We take out the balls at random from the bin without replacement until there is either no red or no blue ball left. With probability $\frac{k}{\ell+k}$ there is no blue ball left, and with probability $\frac{\ell}{\ell+k}$ there is no red ball left.*

PROOF: Let us modify the model a little bit. Instead of taking out the balls until there is either no red or no blue ball left, we take out the balls at random from the bin without replacement until there is no ball left in the bin. The color of the last ball taken out of the bin clearly determines the ball color that has been removed at the time where there is only a single color left. Since each of the $\binom{\ell+k}{k}$ orders of taking out all balls is equally likely and there are $\binom{\ell+k-1}{k}$ orders in which the last ball taken out is blue, the probability that the last ball is blue is

$$\binom{\ell+k-1}{k} \Big/ \binom{\ell+k}{k} = \frac{(\ell+k-1)!\,\ell!\,k!}{k!\,(\ell-1)!\,(\ell+k)!} = \frac{\ell}{\ell+k}.$$

This proves the lemma. □

Using this lemma we are now able to prove Theorem 4.

PROOF OF THEOREM 4: In the phase until the larger or the smaller vertex set are chosen completely by RLS, only steps that increase the number of vertices are accepted. This is because a reduction of the number of vertices in this phase also reduces the number of covered edges and thus the fitness value. Moreover, if the larger vertex set is the vertex set that is first determined completely by RLS, there is no chance for RLS to determine the optimal solution, since only steps that reduce the number of vertices in the smaller vertex set are accepted. In this situation, the optimization time is infinite. Therefore, we have to prove that this happens with positive probability.

For this purpose, we like to apply Lemma 1. But this is not possible in a direct way because of the initialization phase in RLS. In order to overcome this obstacle, we model the initialization phase in the following way. Instead of choosing every vertex with probability 1/2, we choose a $k \in \{0, 1, \ldots, n\}$ following the binomial distribution $B(n, 1/2)$. In other words, we choose k with probability $\binom{n}{k}(\frac{1}{2})^n$. Afterward, we choose successively k of the n vertices without repetition. In order to justify this model, we

have to show that the number of chosen vertices in this model has the same probability distribution as in the real model of the initialization phase, and that each vertex is chosen with probability 1/2 in this model. The probability that we choose exactly k balls in the new model is given as $\binom{n}{k}(\frac{1}{2})^n$ as in the normal initialization phase. And the probability for every vertex to be chosen as one of the k balls is clearly $\frac{k}{n}$. Thus, the probability for each ball to be chosen in the new model is

$$\sum_{k=0}^{n} \frac{k}{n}\binom{n}{k}\left(\frac{1}{2}\right)^n = \sum_{k=1}^{n} \frac{k}{n}\binom{n}{k}\left(\frac{1}{2}\right)^n = \sum_{k=1}^{n} \binom{n-1}{k-1}\left(\frac{1}{2}\right)^n = \frac{1}{2}.$$

Hence, we have justified this model, and we can assume that, starting with the empty subgraph, all vertices are chosen successively with equal probability. We can apply Lemma 1. (Instead of taking out a ball, we choose a vertex that was not chosen so far.) Therefore, the probability that the larger set of vertices is the first set that is completely chosen by RLS is ε. This proves the theorem. □

Theorem 4 shows that the approximability of RLS for the vertex cover problem can be arbitrarily bad. Choosing, for example, $\varepsilon = 1/n$, leads to a graph where V_1 consists of one single vertex. In this case, RLS does not obtain an approximation better than a factor of $n - 1$ with probability $1/n$. Note that an approximation of almost that quality can be obtained for an arbitrary graph by choosing all vertices of the given input.

Now we consider the behavior of the (1+1) EA on the graph B. After having obtained the vertex set V_2 and discarding the set V_1, the (1+1) EA cannot obtain a better approximation ratio than $(1 - \varepsilon)/\varepsilon$ without flipping at least εn bits. If ε is not too small, the (1+1) EA can only leave this local optimum in the next mutation step with a probability that is exponentially small. Therefore, the expected optimization time under the condition that such a solution has been produced before having obtained the optimal solution is exponential. The following theorem shows that this can lead to almost arbitrarily bad approximation ratios of roughly $n^{1-\delta}$, where $\delta > 0$ is a constant.

THEOREM 5: *Let $\delta > 0$ be a constant and $n^{\delta-1} \leqslant \varepsilon < 1/2$. The expected optimization time of the (1+1) EA on B (with $|V_1| = \varepsilon n$ and $|V_2| = (1 - \varepsilon)\, n$) is exponential. Moreover, the expected time to produce an approximation better than a factor $(1 - \varepsilon)/\varepsilon$ is exponential.*

PROOF: We investigate a run of two phases. In the first phase, we examine the probability that a vertex cover including all vertices of V_2 with at least one vertex missing in V_1 is constructed. In the second phase, we give a lower bound for the probability that a local optimum is obtained by removing all vertices of V_1. This local optimum can only be left by including all vertices of V_1 and removing at least εn vertices of V_2.

The first phase consists of $12en \ln n$ mutation steps. First we prove that the (1+1) EA obtains a vertex cover including all vertices of V_2 within this phase with probability at least 1/4. We consider only the effect of steps that flip exactly one bit in V_2 and no other bit; these steps are called *simple V_2 steps* in the following. The probability for a simple V_2-step is $((1 - \varepsilon)\, n)\,(1/n)\,(1 - 1/n)^{n-1} \geqslant (1 - \varepsilon)/e$. Thus, the average wait time for such a simple V_2-step is at most $e/(1 - \varepsilon)$. We apply Markov's inequality on the wait time for $k\,(1 - \varepsilon)/(2e)$ such steps. Hence, with probability at least 1/2, there are in k steps of the (1+1) EA at least $k\,(1 - \varepsilon)/(2e)$ special V_2 steps. Using $1 - \varepsilon \geqslant 1/2$, this means that the considered phase of $12en \ln n$ mutation steps contains with probability at least 1/2

at least $3n \ln n$ such simple V_2 steps. Considering this number, we apply the method of expected multiplicative weight decrease in a more precise way than in Theorem 1, where weight denotes the number of uncovered edges.

Let N be the current number of uncovered edges. All simple V_2 steps that add a vertex of V_2 are accepted and the total weight decrease of these steps is N, since the choice of V_2 is clearly a valid vertex cover. Simple V_2 steps removing vertices of V_2 contribute a weight decrease of 0. There are, altogether, $(1-\varepsilon)\,n$ simple V_2 steps. Thus, a simple V_2 step decreases the number of uncovered edges by an expected factor of $1-1/((1-\varepsilon)n) \leqslant 1-1/n$. Since $N \leqslant n^2$, the expected number of uncovered edges after t simple V_2 steps is at most $(1-1/n)^t \cdot n^2$. Assuming $3n \ln n$ such steps, the expected number of uncovered edges after the phase is at most $n^2\,(1-1/n)^{3n \ln n} \leqslant 1/n$, which is strictly less than 1/2. Hence, using Markov's inequality and the bound 1/2 on the probability of having enough simple V_2 steps, a cover is produced with probability at least $(1/2) \cdot (1/2) = 1/4$ in this phase.

Now we prove a lower bound on the probability that after $12en \ln n$ steps of the (1+1) EA, at least one vertex of V_1 has not been chosen. This is exactly the case if the (1+1) EA completely discovers V_2 before completely discovering V_1. We base the analysis on the assumption that $12en \ln n$ steps lead to a vertex cover including all vertices from V_2 and note that this assumption does not decrease the probability of an unchosen vertex from V_1. By Chernoff bounds, there are with probability $1-2^{-\Omega(\varepsilon n)} = 1-2^{-\Omega(n^\delta)}$ at least $|V_1|/3 = \varepsilon n/3 \geqslant n^\delta/3$ unchosen vertices in V_1 in the initial solution. The probability that after $12en \ln n$ mutation steps of the (1+1) EA, a single vertex is chosen at least once is $1-(1-1/n)^{12en \ln n}$. Thus, the probability that at least one of the initially not chosen vertices of V_1 is not chosen after $12en \ln n$ mutation steps of the (1+1) EA is

$$1-\left(1-\left(1-\frac{1}{n}\right)^{12en \ln n}\right)^{\frac{n^\delta}{3}} \geqslant \frac{n^{\delta-13e}}{6}.$$

Altogether, the probability that the (1+1) EA chooses all vertices of V_2 before choosing all vertices of V_1 is bounded from below by $(n^{\delta-13e}/6) \cdot (1/4) = n^{\delta-13e}/24$. Hence, the probability is, as wanted, at least bounded by an inverse polynomial from below.

We consider a second phase of $n^{3/2}$ mutation steps and show that all vertices of V_1 are removed with probability at least 1/15. Let us assume that we start this phase with all vertices of V_2 and all but one vertex of V_1 in the current solution. This is the worst case for our analysis. In this phase (all vertices of V_2 and some vertices of V_1 chosen) the only mutation steps accepted by the (1+1) EA are the following. Either all missing vertices of V_1 are chosen and at least as many vertices of V_2 are removed ("bad event"), or all vertices of V_2 are kept and the number of vertices in V_1 is decreased or stays the same by adding and removing some vertices ("good event"). The former mutation step has a probability of at most n^{-k}, where k denotes the current number of missing vertices in V_1. For the latter kind of mutation steps, we restrict ourselves to 1-bit flips reducing the number of vertices in V_1. The probability for such a mutation step is at least $\frac{\varepsilon n-k}{en} \geqslant \frac{1}{en}$. For our calculations, we take only those two kinds of mutation steps into account, the good event with probability at least $\frac{\varepsilon n-k}{en}$ and the bad event with probability at most n^{-k}, since all other accepted mutation steps reduce or preserve the number of vertices in V_1. The probability that the good event occurs before the bad event is at least $\frac{1}{en}/(\frac{1}{en}+n^{-k}) = 1-\frac{e}{n^{k-1}+e}$. Thus, the probability that the vertices of V_1 were all removed

by the (1+1) EA before the bad event occurs is at least

$$\prod_{k=1}^{\varepsilon n-1}\left(1-\frac{e}{n^{k-1}+e}\right) \geqslant \frac{1}{1+e}\left(1-\frac{e}{n}\right)^{\frac{n-1}{2}} \geqslant \frac{e^{-e/2}}{1+e} > \frac{1}{15}.$$

The expected waiting time for removing all vertices of V_1 by the (1+1) EA is $\mathcal{O}(n \log n)$ and therefore all vertices of V_1 are removed within $n^{3/2}$ steps with probability $1 - o(1)$ using Markov's inequality (always assuming that the bad event does not occur during this phase). Hence, the probability that the (1+1) EA determines the local minimum V_2 as the vertex cover is at least $\frac{n^{\delta-13e}}{360}$. But if the current solution is V_2, every accepted mutation step has to add all the vertices of V_1 (and remove at least $|V_1|$ vertices of V_2). This occurs with probability at most $n^{-\varepsilon n} = n^{-\Omega(n^\delta)}$. Thus, the expected time until an approximation better than a factor $(1-\varepsilon)/\varepsilon$ is determined is at least

$$\frac{n^{\delta-13e}}{360} n^{\Omega(n^\delta)} = n^{\Omega(n^\delta)}.$$

This proves the theorem. □

The preceding theorem proves that a simple (1+1) EA gets stuck on the bipartite graph class with at least constant probability. Multi-start variants of the algorithms can improve the success probability drastically (Oliveto et al., 2007) while the straightforward use of populations in single-objective formulations does not necessarily allow for a significant increase in success probability (Oliveto et al., 2008). In contrast to the single-objective formulations, SEMO and global SEMO have the ability to overcome this obstacle. The main reason for this is that the multi-objective model makes the algorithm behave in a greedy way. Note that each vertex of V_1 is incident to $(1-\varepsilon)n$ edges while each vertex of V_2 is incident to εn edges. A greedy algorithm that starts with the empty vertex set and adds in each step a vertex which covers a largest number of up to now uncovered edges ends up with V_1 and therefore produces an optimal solution. It is well known that many covering problems can be approximated with ratio $\log n$ using algorithms of that kind.

THEOREM 6: *The expected optimization time of SEMO and global SEMO on B is $\mathcal{O}(n^2 \log n)$.*

PROOF: We prove the theorem for global SEMO. All subsets of V_1 are Pareto optimal. The objective vector of a subset $V' \subseteq V_1$ with $|V'| = k$ is $(m - k(1-\varepsilon)n, k)$. The Pareto front contains the $|V_1| + 1 = \varepsilon n + 1$ objective vectors $(m, 0), (m - (1-\varepsilon)n, 1), (m - 2(1-\varepsilon)n, 2), \ldots, (0, \varepsilon n)$, where $m = \varepsilon(1-\varepsilon)n^2$. The population size is bounded by $\mathcal{O}(n)$, because a population can never contain two individuals with an equal number of vertices.

First, we determine the time until the Pareto optimal search point $(m, 0)$ is found. Since it is the only one with $|x|_1 = 0$, it is never removed from the population again. One possibility for global SEMO to get "closer" to $(m, 0)$ is to select the individual with the smallest $|x|_1$-value from the current population and mutate it so that the $|x|_1$-value decreases. By the coupon collector's theorem (Motwani and Raghavan, 1995), this shows that $(m, 0)$ is included in the population after $\mathcal{O}(n^2 \log n)$ steps with high probability since the population size is bounded by $\mathcal{O}(n)$.

We now bound the time to discover the whole Pareto set after $(m, 0)$ is found. Since the probability of flipping a single bit in one step is at least $1/e$, the probability to get from one Pareto optimal solution $(m - k\,(1-\varepsilon)\,n, k)$ to the "next" Pareto optimal solution $(m - (k+1)\,(1-\varepsilon)\,n, k+1)$ is $(\varepsilon n - k)/(e\,n)$. Again using the linear size of the population, the expected number of steps to gain the whole Pareto front is at most $\sum_{k=0}^{\varepsilon n-1}(e\,n^2)/(\varepsilon n - k) = \mathcal{O}(n^2 \log n)$, which completes the proof. As only 1-bit flips are used in the proof, the result also holds for SEMO. □

4 The SetCover Problem

As a generalization of the VertexCover problem we consider the well-known SetCover problem and examine the approximation ability of the multi-objective and the single-objective approach. Given a ground set $S = \{S_1, \ldots, S_m\}$ and a collection $C_1, \ldots, C_n$ of subsets of S with corresponding positive costs $c_1, \ldots, c_n$. We denote by $c_{\max} = \max_i c_i$ the maximum cost of a subset for a given instance. The goal is to find a minimum-cost selection $C_{i_1}, \ldots, C_{i_k}$, $1 \leqslant i_j \leqslant n$ and $1 \leqslant j \leqslant k$, of subsets such that all elements of S are covered. The SetCover problem cannot be approximated better than by a factor of $\log m$ unless certain assumptions from complexity theory do not hold (Raz and Safra, 1997; Feige, 1998). It is well known that Chvatal's simple greedy algorithm (Chvatal, 1979) achieves a worst-case approximation ratio of $\mathcal{O}(\log m)$. In the following, we want to strengthen our claim that a multi-objective model might be superior to a corresponding single-objective approach as it has the ability to simulate a greedy approach.

Considering the algorithms introduced in Section 2, a search point $x \in \{0,1\}^n$ encodes a selection of subsets. $p(x) = \sum_{i=1}^{n} c_i x_i$ measures the total cost of the selection and $u(x)$ denotes the number of elements of S that are uncovered. Considering RLS and the (1+1) EA for the SetCover problem, the fitness of a search point x is given by the vector $f(x) = (u(x), p(x))$ which should be minimized with respect to the lexicographic order. In our multi-objective setting, we would like to minimize $u(x)$ and $p(x)$ at the same time.

We start by showing that with high probability, the RLS and the (1+1) EA are not able to compute solutions that achieve more than a trivial approximation ratio. This is done by generalizing our negative results for the single-objective approach of the previous section to the SetCover problem. The VertexCover problem for a given graph $G = (V, E)$ is a special SetCover problem where $S = E$ and C_i denotes the set of edges incident to vertex v_i and $c_i = 1$ for $i \in \{1, \ldots, n\}$.

We consider a generalization of the graph B given in the previous section to the SetCover problem and show that the approximation ratio achievable by the single-objective algorithms can be unbounded. The idea is to consider subsets C_i, $1 \leqslant i \leqslant n$, that correspond to the set of edges incident to the different vertices of B and assign large costs to subsets corresponding to vertices in V_2 and small costs corresponding to vertices in V_1. We make this precise and denote our class of instances by C^*. Let

$$\begin{aligned} S = \{&\{v_1, v_{\varepsilon n+1}\}, \ldots, \{v_1, v_n\},\\ &\{v_2, v_{\varepsilon n+1}\}, \ldots, \{v_2, v_n\},\\ &\ldots\\ &\{v_{\varepsilon n}, v_{\varepsilon n+1}\}, \ldots, \{v_{\varepsilon n}, v_n\}\} \end{aligned}$$

be the ground set, where

$$C_i = \{\{v_i, v_{\varepsilon n+1}\}, \ldots, \{v_i, v_n\}\}$$

with $c_i = 1$, $1 \leqslant i \leqslant \varepsilon n$, and

$$C_k = \{\{v_k, v_1\}, \ldots, \{v_k, v_{\varepsilon n}\}\}$$

where $c_k = c_{\max}$, $\varepsilon n + 1 \leqslant k \leqslant n$, are the subsets with associated costs, where $c_{\max} \geqslant 1$. We assume that $c_{\max}$ is a large value (e.g., $c_{\max} = 2^n$) to show that the approximation achievable by RLS and the (1+1) EA in expected polynomial time may be arbitrarily bad.

In the proofs of Theorems 4 and 5, we examine the probability that RLS and the (1+1) EA obtain the larger partition of the bipartite graph before the smaller one. As long as a vertex cover has not been obtained, each mutation step that decreases the number of uncovered edges is accepted. We can translate the arguments given in the proofs to the SETCOVER instances. Vertices in the graph B are mapped to sets of C^*. Again, each mutation step reducing the number of uncovered elements is accepted and the probability of choosing the sets C_i, $1 \leqslant i \leqslant \varepsilon n$, before the sets C_j, $\varepsilon n + 1 \leqslant j \leqslant n$, can be bounded in the same way as for the graph B. Therefore, we can generalize Theorems 4 and 5 to C^* in the following way.

THEOREM 7: *With probability ε, RLS cannot obtain an approximation better than a factor $((1-\varepsilon)c_{\max})/\varepsilon$ for C^* within a finite number of steps. Moreover, the expected time to produce an approximation better than a factor $((1-\varepsilon)c_{\max})/\varepsilon$ on C^* is infinite.*

THEOREM 8: *Let $\delta > 0$ be a constant and $n^{\delta-1} \leqslant \varepsilon < 1/2$. The expected optimization time of the (1+1) EA on C^* (with $|V_1| = \varepsilon n$ and $|V_2| = (1-\varepsilon)n$) is exponential. In particular, the expected time to produce an approximation better than a factor $((1-\varepsilon)c_{\max})/\varepsilon$ is exponential.*

Theorems 7 and 8 show that the approximation quality achievable in expected polynomial time can be made arbitrarily bad as long as $c_{\max}$ grows. We therefore say that RLS and the (1+1) EA have a worst case approximation ratio obtainable in expected polynomial time for the SETCOVER problem that is unbounded. In contrast to this, we show that the expected optimization time of SEMO and global SEMO on C^* is polynomial. The following properties hold for the multi-objective model of the SETCOVER problem. The all-zeros string is Pareto-optimal since it covers no elements at zero cost. Moreover, any population of the multi-objective algorithms, which is a set of mutually nondominating search points, can have at most $m + 1$ elements.

THEOREM 9: *The expected optimization time of SEMO and global SEMO on C^* is $\mathcal{O}(mn(\log c_{max} + \log n))$.*

PROOF: In order to prove the theorem, we generalize some ideas already used in the proof of Theorem 6. The Pareto front consists of the objective vectors $(m, 0)$, $(m - (1-\varepsilon)n, 1)$, $(m - 2(1-\varepsilon)n, 2)$, $(0, \varepsilon n)$. A solution corresponding to the objective vector $(m - i(1-\varepsilon)n, i)$, $1 \leqslant i \leqslant \varepsilon n$, chooses exactly i subsets from the set $\{C_1, \ldots, C_{\varepsilon n}\}$ of subsets with costs 1. We first consider the time until the search point 0^n with Pareto optimal objective vector $(m, 0)$ has been included into the population.

In order to estimate this time, we consider the expected multiplicative decrease of the minimum p-value for the current population. The probability of choosing an individual with minimum p-value among all individuals in the population is $\Omega(1/m)$ as the population size is bounded above by $m+1$. Since flipping a single bit decreases the p-value by an expected factor of $1-1/(en)$ or better, the expected time until the all-zeros string is reached is bounded above by $\mathcal{O}(mn\,(\log c_{\max}+\log n))$.

After having obtained a Pareto optimal solution x with objective vector $(m-k(1-\varepsilon)\,n,k)$, $0\leqslant k<\varepsilon n$, there are $\varepsilon n-k$ subsets of costs 1 that can be chosen to obtain a Pareto-optimal solution whose objective vector is $(m-(k+1)(1-\varepsilon)\,n,k+1)$. Taking into account the upper bound on the population size as well as flipping one of the desired bits in x, the probability that such a step happens in the next iteration is at least $\frac{\varepsilon n-k}{enm}$. Hence, the expected time to obtain for the "next" Pareto optimal objective vector a corresponding solution is upper bounded by $\mathcal{O}((mn)/(\varepsilon n-k))$. Summing up over the different values of k a solution for each Pareto optimal objective vector has been produced after an expected number of $\mathcal{O}(mn\log n)$ steps under the condition that the search point 0^n has been obtained before, which completes the proof. □

Up to now, we have pointed out classes of problems where the multi-objective approach achieves better approximations than the single-objective one. We have also shown that the single-objective algorithms can only achieve a trivial approximation ratio within an expected polynomial number of steps. In contrast to this, we point out in the following that the multi-objective model leads to good approximations within an expected polynomial number of steps. Here, we are in particular interested in the expected number of steps until a solution x with $u(x)=0$ has been produced that is a good approximation of an optimal one.

We will show that SEMO and global SEMO are able to efficiently find approximate solutions to arbitrary instances of the NP-hard SETCOVER problem. The approximation quality is, up to a constant factor, the best we can hope for in polynomial time for arbitrary instances, unless P=NP (Raz and Safra, 1997).

THEOREM 10: *For any instance of the* SETCOVER *problem and any initial search point, SEMO and global SEMO find an* $(\ln(m)+1)$*-approximate solution in an expected number of* $\mathcal{O}(m^2n+mn\,(\log n+\log c_{\max}))$ *steps.*

PROOF: The proof idea is to show that SEMO is able to proceed along the lines of the greedy algorithm for SETCOVER introduced by Chvatal (Chvatal, 1979; Vazirani, 2001). Let $H_m:=\sum_{i=1}^{m}1/i$ be the mth harmonic number and let $R_k:=H_m-H_{m-k}$, $0\leqslant k\leqslant m$, be the sum of the last k terms of H_m. While the greedy algorithm is able to find H_m-approximate solutions, SEMO creates R_k-approximate solutions that cover k elements for increasing values of k, that is, it arrives at intermediate solutions that are at least as good as in the greedy algorithm. The expected time until the all-zeros string is reached is bounded above by $\mathcal{O}(mn\,(\log c_{\max}+\log n))$ using the same ideas as in the proof of Theorem 9.

Let OPT be the cost of an optimal solution. Let $c(x)=m-u(x)$ be the number of elements of S covered in a solution x. The remainder of the proof studies the so-called potential of the current population, which is the largest k such that there is an individual x in the population where $c(x)=k$ and $p(x)\leqslant R_k\cdot\mathrm{OPT}$. The potential is well defined since now the all-zeros string is always in the population.

It is easy to see that the potential cannot decrease. We examine the expected time until the potential increases at least by 1. To this end, we apply the analysis of the greedy algorithm by Vazirani (2001) and use the notion of cost-effectiveness of a set, defined as the cost of the set divided by the number of newly covered elements. If there are $n - k$ elements left to cover and we add the most cost-effective set to cover some of these, all newly covered elements are covered at a relative cost of at most $\mathrm{OPT}/(n - k)$. Hence, if the cost of the selection was bounded above by $R_k \cdot \mathrm{OPT}$ before and $k' \geqslant k + 1$ elements are covered after the step, the cost is at most $R_{k'} \cdot \mathrm{OPT}$ afterward. The probability of choosing an individual that defines the current potential is bounded below by $\Omega(1/m)$. The probability of adding a most cost-effective set is bounded below by $1/(en)$ as it suffices to flip a certain bit. Since the potential can increase at most m times, the expected time is $\mathcal{O}(m^2 n)$ until an R_m-optimal, that is, H_m-optimal, individual covering all elements is created. □

5 Conclusions

The general purpose of randomized search heuristics is to compute good approximations within a small amount of time. While there have been many experimental results reported, only a few theoretical investigations have been carried out up to now. We have investigated the approximation ability of randomized search heuristics for the important class of covering problems. Comparing single-objective and multi-objective models, our results show that the multi-objective model can lead to a better approximation ability of randomized search heuristics. A reason for this may be that the multi-objective approach has the ability to act in a greedy way. In the case of the VERTEXCOVER problem we have pointed out situations where this greedy behavior can make a difference between obtaining optimal solutions and the inapproximability within an expected polynomial number of steps. For the SETCOVER problem, we have shown that randomized search heuristics using a multi-objective model are able to compute a factor $\mathcal{O}(\log m)$-approximation which is the best possible while the use of a single-objective one has a worst case approximation ratio within an expected polynomial number of steps that is unbounded.

Acknowledgments

C. Witt and J. He gratefully acknowledge financial support by the Deutsche Forschungsgemeinschaft (DFG, SFB 531) and by the UK Engineering and Physical Research Council under Grant No. EP/C520696/1, respectively.

References

Chvatal, V. (1979). A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to algorithms* (2nd ed.). Cambridge, MA: MIT Press.

Diestel, R. (2005). *Graph theory* (3rd ed.). Berlin: Springer.

Droste, S., Jansen, T., and Wegener, I. (2002). On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276:51–81.

Feige, U. (1998). A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652.

Giel, O. (2003). Expected runtimes of a simple multi-objective evolutionary algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '03)*, pp. 1918–1925.

Giel, O., and Wegener, I. (2003). Evolutionary algorithms and the maximum matching problem. In *Proceedings of the International Symposium on Theoretical Aspects of Computer Science (STACS '03)*, Vol. 2607 of *LNCS*, pp. 415–426.

Glaser, O. (2005). Evolutionäre Algorithmen und das Vertex Cover Problem (in German). Department of Computer Science, University of Kiel, Studienarbeit.

He, J., Yao, X., and Li, J. (2005). A comparative study of three evolutionary algorithms incorporating different amounts of domain knowledge for node covering problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 35(2):266–271.

Jansen, T., and Wegener, I. (2001). Evolutionary algorithms—How to cope with plateaus of constant fitness and when to reject strings of the same fitness. *IEEE Transactions on Evolutionary Computation*, 5(6):589–599.

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

Motwani, R., and Raghavan, P. (1995). *Randomized algorithms*. Cambridge, UK: Cambridge University Press.

Neumann, F. (2004). Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. In *Proceedings of the International Conference Parallel Problem Solving from Nature (PPSN '04)*, Vol. 3242 of *LNCS*, pp. 80–89.

Neumann, F., and Wegener, I. (2004). Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. In *Proceedings of the Conference on Genetic and Evolutionary Computation (GECCO '04)*, Vol. 3102 of *LNCS*, pp. 713–724.

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

Oliveto, P. S., He, J., and Yao, X. (2007). Evolutionary algorithms and the vertex cover problem. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 1870–1877.

Oliveto, P. S., He, J., and Yao, X. (2008). Analysis of population-based evolutionary algorithms for the vertex cover problem. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 1563–1570.

Raz, R., and Safra, S. (1997). A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the ACM Symposium on Theory of Computing (STOC '97)*, pp. 475–484.

Vazirani, V. (2001). *Appromixation algorithms*. Berlin: Springer.

Witt, C. (2005). Worst-case and average-case approximations by simple randomized search heuristics. In *Proceedings of the International Symposium on Theoretical Aspects of Computer Science (STACS '05)*, Vol. 3404 of *LNCS*, pp. 44–56.