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

Tobias Friedrich* Jun He† Nils Hebbinghaus* Frank Neumann* Carsten Witt‡

* Algorithms and Complexity Group
Max-Planck-Institut für Informatik
Saarbrücken, Germany

† School of Computer Science
University of Birmingham
Birmingham, United Kingkom

‡ LS2, Fachbereich Informatik
University of Dortmund
Dortmund, Germany

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 ones 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 even no good approximation can be achieved within expected polynomial time. Examining the more general SetCover problem we show that optimal solutions can be approximated within a factor of $\log n$, where n is the problem dimension, using the multi-objective approach while the approximation quality obtainable by the single-objective approach in expected polynomial time may be arbitrarily bad.

Categories and Subject Descriptors: F.2 [Theory of Computation]: Analysis of Algorithms and Problem Complexity

General Terms: Theory, Algorithms, Performance

Keywords: Combinatorial Optimization, Covering Problems, Multi-Objective Optimization, Runtime Analysis

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 within a small amount of time good approximations of optimal solutions. In contrast to their success reported in several applications, there are only a few rigorous results on the approximation ability of randomized search heuristics [15]. Our aim is to study the following question. Is it possible that a multi-objective model of 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 [13], 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 multi-objective evolutionary algorithms (EAs). 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, we assume that in the worst case the process is slowed down by a polynomial factor. In contrast to this, the multi-objective model admits to direct the search in a better way as shown in [13]. In particular multi-objective models may make randomized search heuristics behave greedily. Greedy algorithms play an important role in the classical design of algorithms [1]. 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 the last years, a lot of progress has been made in analyzing simple evolutionary algorithms with respect their runtime behavior on artificial pseudo-boolean functions [3, 8] as well as some well-known combinatorial optimization problems [12, 13, 5, 15,

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

11]. 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. 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 one 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 $n^{1-\delta}$, 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. Afterwards we consider the more general SetCover problem which is hard to approximate within a multiplicative factor better than $\log n$ [14]. For the single-objective approach, we show that the approximation ratio obtainable in expected polynomial time is unbounded. In contrast to this non-approximability result for the single-objective approach, we point out that the multi-objective model leads to a factor $O(\log n)$-approximation for the SetCover problem which is best we can hope for under certain assumptions from complexity theory [14].

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 $O(\log n)$-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 [4, 9] and for different kinds of spanning tree problems [11, 13]. It 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 iff 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 iff $f(y) \leq f(x)$ and $f(y) \neq f(x)$. $f(y) \leq f(x)$ holds iff $f_i(y) \leq f_i(x)$ 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 $f(x') \leq f(z)$ 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 $f(x') \leq f(x)$, 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 one of the well-known NP-hard combinatorial optimization problems. 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. [1]). 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. [2]). In this case the number of edges in a maximum matching equals the number of vertices in a minimum vertex cover.

Considering the algorithms of Section 2 for the VertexCover problem, 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 $x_i = 1$ and otherwise it is unchosen. We use the fitness function considered by He, Yao, and Li [7]. 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. Afterwards 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. In the case of SEMO and Global SEMO both objectives should be optimized at the same time.

We compare RLS with SEMO and the (1+1) EA with Global SEMO by runtime analyses. He, Yao, and Li [7] have already examined a single objective EA 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. They have shown that their algorithm finds a vertex cover in a number of $O(n^2)$ generations. We show that the expected time until RLS and the (1+1) EA have produced a vertex cover is $O(n \log n)$. A similar proof can be found in [6]. In addition, we show that this bound is tight by presenting a worst case example. As there is always a constant probability, in the EA analyzed by He, Yao, and Li [7], to use only mutation the upper bound of $O(n \log n)$ also holds in their scenario.

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

Proof. We prove the theorem for the (1+1) EA using the method of the expected multiplicative weight decrease developed in [12]. 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. Non-accepted 1-bit flips contribute a value of 0 to the reduction of the number of uncovered edges. We consider the expected decrease 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 $t^* = cn \log n$, c 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 $2t^* = O(n \log n)$ (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 [10]. 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 is produced after an expected number of $O(n \log n)$ steps. This solution is either a minimum vertex cover (contains exactly $n - 1$ vertices) or a non optimal one (containing n vertices). In the second case, exactly one arbitrary bit has to flip. The expected waiting 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 [10]. As at least $n - 1$ vertices are contained in each vertex cover

at least $n/3-1$ bits have to flip. The probability of non flipping one of these $n/3-1$ bits during $c\,n\log n$ steps, c an appropriate constant, is bounded from below by a positive constant using the ideas of the coupon collector's theorem [3, 10], 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 adding sequentially 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 $O(n)$ 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. The initial search point consists of at most $\frac{2}{3}\cdot n$ vertices with high probability using Chernoff bounds [10]. Let $a_{\max}$ denote the maximal number of vertices of one element in the current population. We consider the time where $a_{\max}\in[\frac{2}{3}n,\frac{3}{4}n]$ 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 $|x|_1=k$, $0\le k\le n-1$ is Pareto optimal and its objective vector is $f(x)=(k,(n-k)(n-k-1)/2)$ as the set of uncovered edges consists of all edges between the unchosen vertices. 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 $a_{\max}\in[\frac{2}{3}n,\frac{3}{4}n]$ is bounded by 2. To obtain from a step that increases $a_{\max}$ by i a step that increases $a_{\max}$ by $i+1$ 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 such steps 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 larger than 4 with probability less than 1/2. It follows that with probability 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. With high probability $a_{\max}$ is less than or equal to $n-2n^{1/4}$ at this time. In other words, 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 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 upper bounded by $e^{-n^{1/4}}$ and therefore such an event does not happen within a polynomial number of steps with probability exponentially close to 1. We call a step a k-step iff it creates a solution z with $|z|_1>|x|_1$ by flipping k 0-bits. The probability to flip k 0-bits in a single mutation step of a solution y with $|y|_0\le r+n^{1/4}$ is upper bounded by $\left(\frac{r+n^{1/4}}{n}\right)^k=O(n^{-3k/4})$ and the probability that a solution y with $r\le|y|_0\le r+n^{1/4}$ is chosen for mutation and a k-step is performed is $O(n^{-3(k+1)/4})$ as the population size is $\Theta(n)$. Hence, for $k\ge 2$ this does not happen within $\Theta(n^2\log n)$ 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 x. The expected time to reduce the value r of zeros to $r-1$ zeros by 1-steps 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 of order

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

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

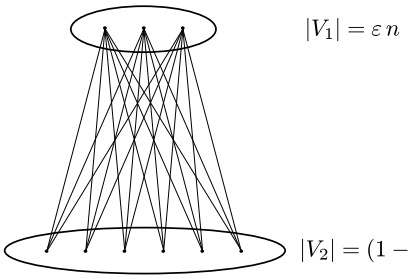

Figure 1. The considered complete bipartite graph $B=(V,E)$ for $n=9$ and $\varepsilon=\frac{1}{3}$.

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 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 non equal 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 $|V_1|=\varepsilon n$ and $|V_2|=(1-\varepsilon)n$, $\varepsilon<1/2$ and 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 can not 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 $(1-\varepsilon)/\varepsilon$ 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 $\frac{k}{l+k}$ there is no blue ball left, and with probability $\frac{l}{l+k}$ 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 firstly removed from the bin. Since every of the $\binom{l+k}{k}$ orders of taking out all balls is equally likely and there are $\binom{l+k-1}{k}$ orders in which the last ball taken out is blue, the probability that the last ball is blue is

$$\tbinom{l+k-1}{k}/\tbinom{l+k}{k} = \tfrac{(l+k-1)!l!k!}{k!(l-1)!(l+k)!} = \tfrac{l}{l+k}.$$

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 reduces also 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 larger 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. 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 $\binom{n}{k}(\frac{1}{2})^n$. Afterwards, we choose successively k of the n vertices without repetition. 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 initialisation 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

$$\sum_{k=0}^{n} \tfrac{k}{n}\tbinom{n}{k}(\tfrac{1}{2})^n = \sum_{k=1}^{n} \tfrac{k}{n}\tbinom{n}{k}(\tfrac{1}{2})^n = \sum_{k=1}^{n} \tbinom{n-1}{k-1}(\tfrac{1}{2})^n = \tfrac{1}{2}$$

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, e. g., $\varepsilon = 1/n$, 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 can not 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 $n^{1-\delta}$, $\delta > 0$ a constant.

THEOREM 5. *Let $\delta > 0$ be a constant and $n^{\delta-1} \leq \varepsilon < 1/2$. 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 within this phase with probability at least 1/4. We restrict ourselves to the effect of 1–bit flips of vertices in V_2. The probability for a 1–bit flip of a vertex in V_2 is $\frac{(1-\varepsilon)n}{n}(1-\frac{1}{n})^{n-1} \geq \frac{(1-\varepsilon)}{e}$. Thus, the average waiting time for such a mutation step is at most $\frac{e}{(1-\varepsilon)}$ and with probability at least 1/2 there are in k steps of the (1+1) EA at least $\frac{k(1-\varepsilon)}{2e}$ 1–bit flips in V_2 by Markov's inequality. Moreover, there are $(1-\varepsilon)n$ of such 1–bit flips concerning V_2 (one for each vertex). We apply the method of expected multiplicative weight decrease in a more precise way than in Theorem 1. Let N be the current number of uncovered edges. All 1–bit flips adding a vertex of V_2 are accepted and the total weight decrease of these steps is N. 1–bit flips removing vertices of V_2 contribute a weight decrease of 0. Thus, flipping a single 0-bit of V_2 decreases the number of uncovered edges edges by an expected factor of $1-\frac{1}{(1-\varepsilon)n} \leq (1-\frac{1}{n})$. Taking into account that the number of uncovered edges is at most $\varepsilon n(1-\varepsilon)n \leq n^2$, the expected number of uncovered edges after k 1–bit flips in V_2 is at most $(1-\frac{1}{n})^k \cdot n^2$. Considering a phase of $12en\ln n$ steps the expected number of uncovered edges after this phase is strictly less than 1/2. Hence, by Markov's inequality, a cover is produced with probability at least $\frac{1}{2}\frac{1}{2} = \frac{1}{4}$ 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. Since all our considerations up to now concerned only vertices in V_2 and all our considerations from now on are concerned only with vertices in V_1, these two events are independent, and we can later on easily estimate the probability that both events occur simultaneously. This is exactly the case if the (1+1) EA completely discovers V_2 before completely discovering V_1. By Chernoff bounds, there are with probability $1-2^{-\Omega(\varepsilon n)} = 1-2^{-\Omega(n^\delta)}$ at least $|V_1|/3 = \varepsilon n/3 \geq n^\delta/3$ 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 $1-(1-\frac{1}{n})^{12en\ln n}$. 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

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

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 $\frac{n^{\delta-13e}}{24}$. 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, 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). 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 $\frac{\varepsilon n-k}{en} \geq \frac{1}{en}$. For our calculations we take only those two kind 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

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

The expected waiting time for removing all vertices of V_1 by the (1+1) EA is $O(n\log n)$ 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 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

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

This proves the theorem. □

In contrast to RLS and the (1+1) EA, 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 produces therefore an optimal solution. It is well-known that many covering problems have worst case approximation ratio $\log n$ using algorithms of that kind.

THEOREM 6. *The expected optimization time of SEMO and Global SEMO on B is $O(n^2\log n)$.*

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 $(m,0)$, $(m-(1-\varepsilon)n, 1)$, $(m-2(1-\varepsilon)n, 2)$, …, $(0, \varepsilon n)$, where $m = \varepsilon(1-\varepsilon)n^2$. The population size is bounded by $O(n)$ as a population can never contain two individuals with 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 such that the $|x|_1$-value decreases. By the Coupon Collector's theorem [10] this shows that $(m,0)$ is included in the population after $O(n^2\log n)$ steps with high probability since the population size is bounded by $O(n)$.

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 $(\varepsilon n-k)/(en)$. Using again the linear size of the population, the expected number of steps to gain the whole Pareto front is at most $\sum(en^2)/(\varepsilon n-k) = O(n^2\log n)$, 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 $C_{i_1}, \ldots, C_{i_k}$, $1 \leq i_j \leq n$ and $1 \leq j \leq k$, of subsets such that all elements of S are covered. The SETCOVER problem can not be approximated better than by a factor $\log n$ unless certain assumptions from

complexity theory do not hold. It is well known that simple greedy algorithms achieve a worst-case approximation ratio of $O(\log n)$. 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 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, $1 \leq i \leq n$, 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,

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

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

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

with $c_k = c_{\max}$, $\varepsilon n + 1 \leq k \leq n$, be the subsets with associated costs, where $c_{\max}$ is a large value (e. g., $c_{\max} = 2^n$).

Theorem 4 can be generalized to the C^* in the following way using the same proof ideas.

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.*

In a similar way, we can adapt Theorem 5 to the instance C^* of the SetCover problem.

Theorem 8. *Let $\delta > 0$ be a constant and $n^{\delta-1} \leq \varepsilon < 1/2$. 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.*

Theorem 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 non-dominating search points, can have at most m elements.

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

Proof. To prove the theorem we generalize some ideas already used in the proof of Theorem 6. The Pareto front consists of the objective vectors $(m, 0), (m-(1-\varepsilon)n, 1), (m-2(1-\varepsilon)n, 2), (0, \varepsilon n)$ and a solution corresponding to the objective vector $(m - i(1-\varepsilon)n, i)$, $1 \leq i \leq \varepsilon n$, 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.

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 $O(mn(\log c_{\max} + \log n))$.

After having obtained a Pareto optimal solution x with objective vector $(m - k(1-\varepsilon)n, k)$, $0 \leq k < \varepsilon n$ 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 $O((mn)/(\varepsilon n - k))$. Summing up over the different values of k a solution for each Pareto optimal objective vector has been produced after an expected number of $O(mn \log n)$ 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.

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

Proof. The proof idea is to show that SEMO is able to proceed along the lines of the greedy algorithm for SetCover [14]. Let $H_m := \sum_{i=1}^{m} 1/i$ be the m-th Harmonic number and $R_k := H_m - H_{m-k}$, $0 \leq k \leq m$, 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, i. e., 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 $O(mn(\log c_{\max} + \log n))$ 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 $p(x) \leq R_k \cdot \mathrm{OPT}$. 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 [14] 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 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 $k' \geq k+1$ elements are covered after the step, the cost is at most $R_{k'} \cdot \mathrm{OPT}$ afterwards. The probability of choosing an individual that defines the current potential is bounded below by $1/m$. 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 $O(m^2 n)$ until an R_m-optimal, i. e., 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. In contrast to many experimental results, 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. The main reason for this is 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 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 $O(\log n)$-approximation which is 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.

Acknowledgements

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

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2. edition, 2001.

[2] R. Diestel. *Graph Theory*. Springer, 3rd edition, 2005.

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

[4] O. Giel. Expected runtimes of a simple multi-objective evolutionary algorithm. In *Proc. of CEC 2003*, IEEE Press, pages 1918–1925, 2003.

[5] O. Giel and I. Wegener. Evolutionary algorithms and the maximum matching problem. In *Proc. of STACS 2003*, volume 2607 of *LNCS*, pages 415–426, 2003.

[6] O. Glaser. Evolutionary algorithms and the vertex cover problem (in German). Department of Computer Science, University of Kiel, 2005.

[7] J. He, X. Yao, and J. Li. 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, 2005.

[8] T. Jansen and I. Wegener. Evolutionary algorithms - how to cope with plateaus of constant fitness and when to reject strings of the same fitness. *IEEE Trans. Evolutionary Computation*, 5(6):589–599, 2001.

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

[10] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[11] F. Neumann. Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. In *Proc. of PPSN 2004*, volume 3242 of *LNCS*, pages 80–89, 2004.

[12] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. In *Proc. of GECCO 2004*, volume 3102 of *LNCS*, pages 713–724, 2004.

[13] F. Neumann and I. Wegener. Minimum Spanning Trees Made Easier Via Multi-Objective Optimization. *Natural Computing*, 5(3):305–319, 2006.

[14] V. Vazirani. *Appromixation Algorithms*. Springer, 2001.

[15] C. Witt. Worst-case and average-case approximations by simple randomized search heuristics. In *Proc. of STACS 2005*, volume 3404 of *LNCS*, pages 44–56, 2005.