On the Analysis of the Approximation Capability of Simple Evolutionary Algorithms for Scheduling Problems

Christian Gunia[*]
Institute of Computer Science, University of Freiburg
Georges-Köhler-Allee-79, 79110 Freiburg, Germany
gunia@informatik.uni-freiburg.de

ABSTRACT

Two of the major difficulties dealing with real-world problems nowadays are their increasing complexity and the decreasing available timespan to create "acceptable" solutions. Due to this and the strongly decreasing costs of CPU-power, non specialized (random) search heuristics gain more and more importance. In this paper we analyze the behavior of two very simple search heuristics on a strongly NP-hard scheduling problem. Although both find feasible solutions in pseudo-polynomial time, at least one of them is not able to present an $(1+\varepsilon)$-approximation for arbitrary $\varepsilon > 0$ with constant probability. Despite this, one of the two presented search heuristics can even compete with a problem-specific algorithm on a certain class of inputs and deliver solutions convergent to optimality for increasing problem size.

Categories and Subject Descriptors: F.2.0 Theory of Computation: Analysis of Algorithms and Problem Complexity: General

General Terms: Algorithms, Performance, Theory

Keywords: Algorithms, Theory, Evolution, Scheduling

1. INTRODUCTION

Although the behavior of deterministic algorithms on approximation problems is well understood and investigated for over 30 years, we know very little about the analysis of the approximation capability of randomized search heuristics like evolutionary algorithms. This seems to change in recent years, maybe in reaction to the growing importance of these algorithms.

We will have a closer look at the problem of minimizing the makespan of a schedule for a set of jobs. Given n jobs with integral processing times w_i and a number m of identical machines, the task is to find an assignment of jobs to the machines to minimize the makespan, i. e., the maximum time a machine is scheduled. The job assignment is done in a non-preemptive manner, i. e., jobs cannot be interrupted but have to run completely at a whole. Maybe this is one of the most studied problems in scheduling theory. In 1966 Graham [7] introduced the probably first worst-case analysis of an approximation algorithm by proving that his list scheduling algorithm obtains a solution in polynomial time, guaranteeing a makespan of not more than $2 - 1/m$ times the optimal makespan. In 1976 Sahni [12] presented an algorithm for the minimal makespan problem on two machines, which obtained a makespan at most $1+\varepsilon$ times the optimal makespan in time $O(n^2/\varepsilon)$ for an arbitrary $\varepsilon > 0$. Therefore, this algorithm is an FPTAS[1] (see [10]). To achieve the same quality on m machines he could only prove time $O(n(n^2/\varepsilon)^m - 1)$ for his presented algorithm. This still is an FPTAS for a fixed number of machines, but since it's exponential in m, it is not one if m is part of the given input. Later on was proven that the minimal makespan problem is NP-hard in the strong sense if m is part of the input and hence, the will be no FPTAS unless P=NP (see [5]). Therefore, the PTAS presented in 1987 by Hochbaum and Shmoys [8] is optimal in some way. They obtained this result by reducing the minimum makespan problem to the closely related bin-packing problem, where n objects with sizes within the interval $[0, 1]$ are given, and have to be put into bins, so that the size of each bin does not exceed 1 and the minimal number of bins is used.

Often (randomized) search heuristics are applied to NP-hard problems, for which no adequate approximation algorithms exist. This is done with the intention not to find the optimal solution, but an approximate one. Nevertheless, almost all results on the complexity of randomized search heuristics (RSHs) are dealing with exact optimization. Doubtless, RSHs like the Metropolis algorithm or Simulated Annealing (see [9]) are applied successfully on real-world problems. Unfortunately, they are currently out of range of rigorous analyses, i. e., without using simplifying models, on most problems. To overcome this issue and to present rigorous analyses of evolutionary algorithms Droste, Jansen and Wegener developed analytic methods (see [3] and [14]). These methods were applied successfully on the analysis of simple evolutionary algorithms on a number of problems including matching problems (Giel and Wegener, [6]) and finding minimal spanning trees (Neumann and Wegener, [11]). Based upon these methods, Witt presented in 2005 an analysis of

[*]This work was done while the author was staying at the University of Dortmund

[1]fully polynomial time approximation scheme

the approximation capability of the (1+1)EA and RLS on the minimum makespan problem for two machines ([15]). In this case the problem is equivalent to an optimization variant of the PARTITION problem, and Witt was able to show that these algorithms find a makespan $4/3$ times the optimal makespan in expected polynomial time, even on worst-case instances. Nevertheless, both algorithms need exponential time to create a solution which provides a makespan at most $4/3 - \varepsilon$ time the optimal makespan for each $\varepsilon > 0$ with high probability. By having a closer look at the average-case approximation time on certain input distributions he showed that the algorithms find solutions, whose expected makespan after a polynomial running time was convergent to optimality for increasing problem size. Moreover, not only the mean values were considered, but he also showed that the corresponding qualities of the presented solutions are obtained with high probability.
Of course, one is interested in generalizing these results to minimum makespan problems with more than two machines. Unfortunately, even by using only three machines one is confronted with "new issues", which cannot occur by using two machines. Based upon the cited paper we generalize the results to situations with a constant number of machines by using a canonical enhancement of Witt's fitness function. To overcome the experienced difficulties we create a new fitness function that is providing the evolutionary algorithm more information and improve the results obtained in the first step. We will also reveal a lower bound on the running time of our algorithm trying to achieve a makespan $(2k/(k+1) - \varepsilon)$ times the optimal makespan on k machines for $\varepsilon > 0$. Finally, by using a third fitness function we are able to transfer one of Witt's average-case analyses to the situation of three machines, asymptotically maintaining his shown bounds for the running time.
In order to simplify the transfer and to make the reading of this paper easier, we take a slightly different look at minimum makespan scheduling problem, just like Witt did. We regard it as a variant of the PARTITION problem. Now the processing times are volumes of the n objects. Scheduling the objects onto k machines means placing the objects into k different bins. The task is to minimize the volume of the – or better of *any* – fullest bin. We will refer to this problem as the k-Partition-Problem for the rest of this paper.
To make the problem accessible to well-known evolutionary algorithms we define this value as a function on the search space $S := \{1, ..., k\}^n$, assuming the i-th component x_i of a search point x to represent the location of object i. The fitness function is defined by $f(x) := \max_i\{w_x(B_i)\}$, where B_i denotes the i-th bin and $w_x(B_i)$ its volume with respect to the search point x. Now we extend the (1+1)EA (e. g., [1]) to the search space S.

Definition 1. [(1+1)EA]

1. Initialize the search point $x = (x_1, ..., x_n)$ uniformly at random.
2. Create x' by independently replacing each component x_i of x with probability $1/n$ by a value drawn uniformly at random from $\{1, ..., k\}\backslash\{x_i\}$.
3. If $f(x') \leq f(x)$, replace x by x'.
4. Continue with Step 2.

We call each cycle of the infinite loop a generation and will always refer to the current search point as x and to the one created by the mutation in Step 2 as x'. This algorithm is able to create every search point of $\{1, .., k\}^n$ within one generation with a positive – but decreasing with growing Hamming-distance – probability. With methods presented by Droste, Jansen and Wegener in [3], it is quite easy to show that the expected running time until an optimal search point is first found is always bounded above by $(kn)^n$, thus, the algorithm solves the given problem exactly in at most expected exponential time. This is in contrast to randomized local search (RLS), which works for S as follows.

Definition 2. [RLS]

1. Initialize the search point $x = (x_1, ..., x_n)$ uniformly at random.
2. Create x' by choosing one component x_i of x uniformly at random and draw its new value uniformly at random from $\{1, ..., k\}\backslash\{x_i\}$.
3. If $f(x') \leq f(x)$, replace x by x'.
4. Continue with Step 2.

Obviously, RLS does a random walk on the search space by moving only to Hamming-neighbors. Thus, it can get stuck in local optima. Both algorithms are sometimes called hill-climbers because of their behavior[2]. Since we are only interested in the fitness value of the current search point after t generations and in the expected number of generations until a certain quality of the current search point is assured, we do not care about the problem of finding an adequate stopping criterion for the infinite loop, which is necessary in application. Although the components of a search point are no bits, we call them bits to stay close to the definition in [1] and call mutations changing i bits an i-bit-mutation.

We want to provide a structural overview of this paper. In this section we introduced the considered algorithms and the NP-hard problem that will be examined, giving a rough overview of the paper. Section 2 deals with a first attempt to form an adequate fitness function and improve this attempt by introducing a "better" fitness function. In Section 3 we will reveal a lower bound for the running time of our algorithms, while Section 4 deals with their average-case behavior on a special class of inputs. This paper closes with some conclusions in Section 5.

2. DEFINITIONS AND METHODOLOGY

Since we are going to talk about approximations, we need a definition of the approximation ratio. We adopt the one used in the literature (e. g., [13]).

Definition 3. [c-approximation] Let $f : S \to \mathbb{R}$ be a fitness function, which is to be minimized and $c \geq 1$. A search point $x \in S$ is called a c-approximation iff $f(x) \leq f_{\min} \cdot c$ holds for $f_{\min}$ denoting the optimal fitness value. Parameter c is called the approximation ratio.

In the following we will show that both algorithms can reach a particular approximation ratio by moving only to Hamming-neighbors. Have a look at one of the fullest bins – say

[2]Normally, these algorithms are used for fitness functions which are to be maximized.

B_f – and one of the emptiest ones – namely B_e. By transferring one object a with volume $w_a \le w^*$ from B_f to B_e the volume of B_f shrinks by w_a. If the difference in their volumes is sufficiently large, the maximum of these two volumes will also reduce by w_a. Thus, we get an intuition that it could be useful to bound the volume of the smallest objects in each fullest bin from above. For technical reasons, w. l. o. g. we expect the volumes to be non-increasingly ordered, i. e., $w_1 \ge w_2 \ge ... \ge w_n \in \mathbb{R}^+$ for the rest of this paper. The sum of all volumes is denoted by w.

Definition 4. [Critical volume] For $k, n \in \mathbb{N}$ with $k \ge 2$, a lower bound l on the optimal f-value for an instance W of the k-Partition-Problem and $1 \le i \le k$, we define the critical volume of bin B_i filled with objects $w_1^i \ge ... \ge w_s^i$ by w_r, where $r := \min\{j | \sum_{m=1}^{j} w_m^i > l\}$.

Let us return to the object a transferred from B_f to B_e. If for our current search point x, we have that $f(x) > l + ((k-1)/k)w^*$ holds, $w_x(B_f) > l + ((k-1)/k)w^*$ will be correct and we will get $w_x(B_e) \le (w - w_x(B_f))/(k-1) < (w - (l + ((k-1)/k)w^*)/(k-1) \le w/k - w^*/k$ by applying the pigeonhole-principle on $w - w_x(B_f)$ and the remaining $k-1$ bins. Therefore, the volume of B_e after the transfer is at most $l + ((k-1)/k)w^*$. Hence, the maximum of the volumes of B_f and B_e was either reduced by w_a or is at most $l + ((k-1)/k)w^*$[3]. This leads to the following lemma.

Lemma 1. *Let W be as in Definition 4. Suppose that from some time on, the critical volume of all bins with respect to each current search point x is bounded above by w^*. RLS [(1+1)EA] working on f reaches a search point x^* with $f(x^*) \le l + ((k-1)/k)w^*$ after $(k-1)(kn)^{k-1} \cdot \lceil w/w_n \rceil$ [$e(kn)^{k-1}\lceil w/w_n \rceil$] more steps in expectation.*

Proof. We use the method of fitness-based partitions (see [14]). Therefore, we partition the search space into fitness levels

$$L_0 := \left\{x \mid f(x) \le l + \frac{k-1}{k} \cdot w^*\right\} \text{ and for } 1 \le i \le \lceil w/w_n \rceil :$$
$$L_i := \Big\{x \mid l + \frac{k-1}{k} \cdot w^* + (i-1) \cdot w_n \le f(x) < l + \frac{k-1}{k} \cdot w^* + i \cdot w_n\Big\}.$$

Note that L_0 consists of the search points which are good enough for our purposes. Since the plus-selection of our two RSHs accepts only successors which are not worse than their parents, the RSH starts in one level L_i and can only move to an L_j with $j \le i$; thus, it moves toward L_0.

If one shows that each level will be left in an expected number of $(k-1)(kn)^{k-1}$ [$e(kn)^{k-1}$] generations, our claim will be proven, due to the additivity of the mean value. We have already seen that a transfer of an (existing) object i with a volume of at most w^* from B_f to B_e reduces the maximum of their volumes by w_i or the maximum drops below $l + ((k-1)/k)w^*$. However, it is easy to create examples in which the f-value does not change at all, because of the existence of another bin as full as B_f. But by repeating our argumentation and transferring at most $k-1$ objects in a row, we can diminish the f-value by at least w_n or it drops below the requested bound. So the RSH is leaving the current fitness level if it has not already reached L_0. Since RLS is only able to perform at most one object transfer per generation, it has to carry out these transfers one at a time. The probability for one particular transfer is $1/(kn)$ and therefore for the at most $k-1$ sufficient transfers at least $(kn)^{-k+1}$. Now we can define phases of length $k-1$ and call them successful iff in each generation one of these transfers is carried out. We already calculated the success-probability for one phase and one successful phase assures leaving the current fitness level, unless L_0 is reached. Since there are no requirements on the beginning of one phase, this success-probability holds for each phase unless the current fitness level is left. Hence, the expected waiting time to leave one fitness level is bounded above by $O((kn)^{k-1})$ phases. This reveals the result for RLS, since one phase takes $k-1$ generations and at most $\lceil w/w_n \rceil$ fitness levels have to be left.

The proof for the (1+1)EA is even easier, because this algorithm can change all needed $k-1$ bits within one generation with probability $e^{-1}(kn)^{-k+1}$ and therefore is waiting at most $e(kn)^{k-1}$ generations in expectation to leave the current fitness level. $\square$

Now we have a powerful tool, helping us to prove the following result.

Theorem 1. *The introduced RSHs working on f find a $(2k/(k+1))$-approximation within $O(wn^{2k-2}/w_n)$ generations in expectation.*

Proof. We distinguish two cases. In the first case $w_1 \le w/(k+1)$ holds. Hence, all volumes are bounded above by this value and so the critical volumes of all bins are. Therefore, we can apply Lemma 1 with $l := w/k$ and $w^* := w/(k+1)$. Now we conclude

$$f(x) \le l + \frac{k-1}{k}w^* = l + \frac{k-1}{k+1} \cdot \frac{w}{k} = l \cdot \frac{2k}{k+1}.$$

Since l is a lower bound for the optimal f-value, the theorem follows for this case.

In the other case $w_1 > w/(k+1)$ holds. Now there are *big* objects whose volume exceeds $w/(k+1)$. We summarize them in $w_1, ..., w_a$ and call the remaining objects *small*. If any bin B contains at least two big objects i and j, its volume is at least $w_i + w_j$ and obviously exceeds $w/(k+1) + \max\{w_i, w_j\}$. On the other hand, due to the pigeonhole-principle there is a bin B' of the remaining $k-1$ bins whose volume is below $(w - w_i - w_j)/(k-1) \le (w - 2w/(k+1))/(k-1) = w/(k+1)$. Hence, a transfer of i or j into B' improves the f-value. Thus, there is a sequence of at most $k-1$ accepted transfers separating the big objects from each other. As long as the big objects stay separated and L_0 is not reached, there will be an object with a volume of at most $w/(k+1)$ in each fullest bin B_f, otherwise, the optimum would have been reached. Now we can adapt our previous argumentation and the fitness levels L_i and will see that at most $2k-2$ appropriate transfers in a row are sufficient to leave the current fitness level[4]. Hence, the introduced RSHs stay only $O((kn)^{2k-2})$ generations on each of the $O(w/w_n)$ fitness levels in expectation. This gives the claim. $\square$

We only mention that by using these methods it can be shown that the same approximation ratio is obtained by

[3]We observe that this may not lead to a decrease of the f-value due to the volumes of the remaining bins.

[4]It takes at most $k-1$ transfers to separate the big objects and $k-1$ transfers to reduce the volume of all fullest bins.

these RSHs within $O(wn/\delta)$ generations in expectation with δ denoting the minimal, positive difference producible between two bins by using the given objects. But the main issue using the fitness function f is the loss of information: compared to the 2-Partition-Problem considered in [15], it only reveals information about the volume of *one* bin, but not of *all* bins. Note, that there can be multiple fullest bins. Hence, we take a look at their number. We have already seen that a transfer of a particular object from the fullest to the emptiest bin means an improvement of the fitness value if there is just *one* fullest bin. Nevertheless, if there are more than one fullest bin, the fitness value does not change at all. Therefore, the above mentioned transfer is reversible – note, that it would not have been reversible if the fitness value had decreased. So, a random walk on the number of fullest bins is possible. As long as i fullest bins exist we can only assure that a special i-bit-mutation improves the fitness value. The expected waiting time for a particular mutation changing i components to occur is $\Omega(n^i)$. This would not adhere a (significant) performance loss if one showed that the random walk "drifts" into the right direction, i. e., it tends to decrease the number of fullest bins instead of increasing it. Unfortunately, we cannot guarantee this. On the contrary, there are situations in which the probability of increasing the number of fullest bins is significantly larger than the probability of decreasing its number. Hence, the random walk slows down the optimization process. We will deal with this problem in the next section by defining a new fitness function, which is preventing this random walk to occur.

3. WORST-CASE ANALYSIS

To compensate the "information loss" going from 2 to $k > 2$ bins, we will establish a fitness function g with a lexicographical-like ordering on its co-domain. To simplify the definition, we first introduce a vectorial function g_{help}. For a search point x the function value $g_{\text{help}}(x)$ is a vector consisting of the volumes of the bins in non-increasing order. Given two search points x and x' we say x' is worse than x – denoted by $x' \succ x$ – if and only if $g_{\text{help}}(x') > g_{\text{help}}(x)$ according to the lexicographical ordering.

Since the considered RSHs use elitist selection the exact fitness values do not matter. Only their ordering is important for selection. Therefore, it is easy to develop a precise and formal description of g following the above statement. But since it is only technical, we will omit it here. Let us take a closer look at the transfer of one object with weight w_i from bin B to B'. We observe that the corresponding search point x' is accepted by the plus-selection in Step 3 iff $w_x(B) - w_i \geq w_x(B')$ holds. We point out that this is always correct, regardless of the position of B and B' within the ordering.

LEMMA 2. *Let W be as in Definition 4. Suppose that from some time on, the critical volume of all bins with respect to each current search point x is bounded above by w^*. RLS working on g reaches a search point x^* with $f(x^*) \leq l + ((k-1)/k)w^*$ after $O(wn/w_n)$ more steps in expectation.*

PROOF. We want to use the method of fitness-based partitions. RLS works on g but we measure the quality of our approximation according to f. Hence, we need to establish a connection between them first. We point out that for all x, x' we have $f(x') > f(x) \Rightarrow g(x') > g(x)$. The main idea using fitness-based partitions is to exploit the fact that the fitness of the current search point cannot decrease during the optimization process. We use the same fitness levels as in the proof of Lemma 1. Since a worsening in the fitness value according to f leads to a worsening in the fitness value according to g, RLS still can move from L_i only to L_j with $j \leq i$ during the optimization process. Hence, it is sufficient to show that each level is left within $O(n)$ generations in expectation to obtain our claim. Assuming that RLS is on Level L_i, each bin's volume is bounded above by $l + ((k-1)/k) \cdot w^* + i \cdot w_n$ according to the fitness levels' definition. Each bin exceeding the volume of $l + ((k-1)/k) \cdot w^* + (i-1) \cdot w_n$ is called critical, and all critical bins with respect to the search point x are summarized in $C(x)$. The idea is that these critical bins prevent the algorithm from reaching the next fitness level. Let us focus on reducing the number of critical bins. Obviously, transferring an object whose volume is bounded above by w^* from a fullest (and correspondingly critical bin) to the emptiest bin, means reducing the number of critical bins by one. If there is no chance of increasing its number by 1-bit-mutations, i. e., by transferring one object, at most $k-1$ of these transfers are sufficient to leave the current fitness level. Hence, the claim is proven, due to the fact that RLS "waits" $O(n)$ generations for the above mentioned "reducing" transfer to happen.

Is there a chance to increase the number of critical bins? Since only transfers from one bin to another bin with a volume at least w_n smaller are accepted, the source-bin of the transfer has to be critical; thus, this bin is not critical after the transfer anymore. At most *one* new critical bin was created, letting the number of critical bins untouched or decreases it by one. Therefore, the number of critical bins cannot be increased by 1-bit-mutations and the claim follows. □

THEOREM 2. *RLS working on g finds a $(2k/(k+1))$-approximation relative to f within $O(wn/w_n)$ generations in expectation.*

PROOF. Again, we are doing a case inspection according to the volume of the largest object. If $w_1 \leq w/(k+1)$ holds, Lemma 2 implies the claim. In the other case, we again have to deal with the big objects $w_1, ..., w_a$. We adopt the fitness levels' former definition and brought already forward the argument that we can do this. Since L_0 consists of the search points which are good enough for our purpose, it is sufficient to show that each L_i is left within $O(n)$ generations in expectation. To obtain this, we apply our knowledge gained by the proof of Theorem 1: there is a sequence of 1-bit-mutations separating the big objects from each other. The problem we are confronted with is the existence of 1-bit-mutations unifying two (already) separated big objects in one bin. We manage this issue by "fixing" the positions of all big objects for a certain period of time, except if their transfer is needed for separation. Since one object is left untouched within one generation with probability $1 - 1/n$ and there are at most k big objects, the probability for their hold-up within $O(n)$ generations is bounded below by

$$\left(1 - \frac{k}{n}\right)^{O(n)} = e^{-k \cdot O(1)} = \Omega(1). \qquad (1)$$

On the other hand, the at most $k-1$ separating 1-bit-mutations occur within an expected number of $O(n)$ generations. This leads to the following definition of a phase. The length of a phase is fixed to $O(n)$ generations – the exact number of generations can easily be determined later. The phase is called successful iff the following conditions are met conjointly.

1. As long as at least two big objects are within one bin, they are separated by an 1-bit-mutation. This has to be done within $O(n)$ generations.

2. The big objects are not moved otherwise within one phase.

3. RLS leaves the current fitness level L_i within $O(n)$ generations.

We already saw that the first and the second condition are met conjointly with at least constant probability, due to the Markov inequality (e. g., [10]) and equation (1). Assuming that they hold, the probability of the third one is also bounded below by a positive constant, since $O(n)$ generations are sufficient for this and the Markov inequality is applicable. Altogether, one cycle through the phase is successful with at least constant probability. Since there are no requirements at the beginning of one phase – except not being in L_0, i. e., not having reached the desired approximation – we can create a sequence of phases, unless the current fitness level is left. The expected number of phases until the current fitness level is left, is bounded above by a constant, since the success-probability is bounded below by a positive constant. This gives the claim. $\square$

Next we want to investigate if it is possible for RLS to reach a significantly better approximation ratio than $2k/(k+1)$ on arbitrary instances. Therefore, we will have a look at the following instance. For a small $0 < \varepsilon \in \mathbb{R}$ there are k big objects of volume $w_g := 1/(k+1) - \varepsilon/(3k)$ and $n-k$ small objects of volume $w_k := (1/(k+1) + \varepsilon/3)/(n-k)$. We call this particular instance W_ε and note that the optimal f-value of $1/k$ can be reached by putting one big and $(n-k)/k$ small objects in each bin – assuming n to be chosen to be a multiple of k. Although we are interested in providing a inapproximability result, we will state a quite simple observation first, which will become useful in the proof of the inapproximability result.

Lemma 3. *Given $k, n \in \mathbb{N}$ and an instance of the k-Partition-Problem with $w_1 = w_2 = ... = w_n$. The expected optimization time of the introduced RSHs working on g is bounded above by $O(n \log n)$ generations.*

Due to space limitations, we will omit this proof. But it can easily be obtained by using fitness-based partitions and transferring objects into the currently emptiest bin. We remark that if each bin is additionally filled with one fixed object of size at most w_1, the RSHs will find a search point x with $\forall(i,j) : |w_x(B_i) - w_x(B_j)| \leq w_1$ within the stated time in expectation.

Theorem 3. *Assuming n to be a multiple of k, the approximation time of RLS working on g for W_ε for a $(2k/(k+1)-\varepsilon)$-approximation with respect to f is infinite with at least constant probability. Hence, the expected number of generations for this is unbounded from below.*

Proof. First, we will construct a situation in which the requested approximation ratio cannot be obtained. After doing this, we will show that RLS reaches this situation with at least constant probability. Let us have a closer look at the approximation ratio. In order to achieve the requested approximation RLS has to find a search point x with $f(x) \leq 2/(k+1) - \varepsilon/k$. As long as two big objects are within one bin, the volume of this bin is at least $2/(k+1) - 2\varepsilon/(3k)$; hence, the desired approximation is not obtained. Now we are showing that RLS reaches with at least constant probability a situation in which two big objects are within one bin, and will not be able to leave it in future. We characterize this search point x^* through the following two conditions.

(C1) There are at least two big objects in B_1 and at least one big object in each bin B_3 to B_k.

(C2) $\forall i \geq 2 : w_{x^*}(B_1) - w_{x^*}(B_i) < w_g$.

Condition (C1) assures that the requested approximation-ratio is not obtained. Since the difference between the volumes of B_1 and *each* other bin B_i is less than w_g through condition (C2), the big objects in B_1 cannot be separated in future, and RLS got stuck.

Now, we turn ourselves to the probability of RLS reaching x^*. Therefore, we define a series of conditions whose fulfillment is leading to the search point x^*. Then we will show that RLS conjointly fulfills them with a probability, which is bounded below by a positive constant. The first condition is to start in the following situation x_0: the big objects are placed as demanded in x^*; additionally at least one small object has to be in each bin $B_1, B_3, ..., B_k$. Since we only arrange the positions of $2k-1$ objects and each object is placed into a particular bin with probability $1/k$ after the random initialization, this constraint holds with probability at least $(1/k)^{2k-1} = \Omega(1)$.

Under the assumption that the RLS starts in x_0, we want it to reach the situation x' characterized by "$w_g < w_x(B_2) \leq w_g + w_k$", letting the $2k-1$ given objects left untouched. Since $2k-1 = \Omega(1)$ holds and each entry of the current search point stays untouched with probability $1/n$ each generation, the $2k-1$ given objects stay untouched for $O(n)$ generations with probability

$$\left(1 - \frac{2k-1}{n}\right)^{O(n)} = e^{-(2k-1)\cdot O(1)} = \Omega(1),$$

i. e., with a positive and constant probability. Therefore, it is sufficient to show, that enough small objects are transferred into B_2 within an expected number of $O(n)$ generations; for such a period of time a constant number of objects or a constant number of components of the search point, resp., will be left untouched with constant probability. The Markov inequality will then provide us with the desired probability.

By regarding

$$\begin{aligned} w_g < w_{x'}(B_2) \quad &\Leftrightarrow \quad w_g < \alpha \cdot w_k \\ &\Leftrightarrow \quad \frac{1}{k+1} - \frac{\varepsilon}{3k} < \alpha \cdot \frac{1/(k+1) + \varepsilon/3}{n-k} \end{aligned}$$

we get

$$\alpha > (n-k) \cdot \frac{1/(k+1) - \varepsilon/(3k)}{1/(k+1) + \varepsilon/3} = (n-k) \cdot (1 - \eta)$$

for some particular $\eta < 1$. Therefore, transferring $(n-k) \cdot (1 - \eta/2)$ small objects into B_2 is sufficient to fill up B_2.

Despite the $2k-1$ fixed objects, there are $n-2k+1$ "free" small objects and obviously

$$(n-k)\cdot(1-\eta/2) < \frac{n-2k+1}{\xi}$$

for some particular $\xi > 1$ is correct if n is sufficiently large. Hence, at every moment there are at least $(n-2k+1) - (n-2k+1)/\xi = \Omega(n)$ potential small objects, which can be transferred into bin B_2. Since the probability of a particular 1-bit-mutation, i. e., a transfer, is $(kn)^{-1}$, the probability of transferring any small object into B_2 is bounded below by $\Omega(1)$. As long as $w_x(B_2) \leq w_g$ holds, no small object can be removed from B_2, since the volume of each other bin is bounded below by w_g. Therefore, x' will be reached within an expected number of $O(n)$ generations.

Let A be the number of small objects within bin B_1 when reaching x'. The volume of the first bin at this point of time is $2\cdot w_g + A\cdot w_k$, and the volume of B_2 obviously somewhere between w_g and $w_g + w_k$. Therefore, the difference between them may be greater than w_g, implying not having reached x^*. After we have shown that within $O(n)$ generations the number of small objects within bin B_1 drops to $A/(k+1)$ – leaving the $2k-1$ given objects untouched – and the transferred $Ak/(k+1)$ small objects are distributed among B_2 to B_k "in a nice way", we see that RLS has reached x^* with at least positive and constant probability.

Since more than the half of all small objects need to be used to fill up B_2 the volume of $B_2, ..., B_k$ is smaller than the volume of B_1, due to the fact that B_1 contains *two* big objects. Hence, a small object removed from B_1 cannot get back into B_1. Since at least $A/(k+1)$ objects are inside B_1, the probability of removing one small object from there is at least $A/((k+1)kn)$. RLS waits for this event an expected number of $kn(k+1)/A$ generations and for all $Ak/(k+1)$ transfers at most $Ak/(k+1)\cdot kn(k+1)/A = k^2n = O(n)$ generations in expectation. Lemma 3 ensures us, that within $O(n)$ generation these $Ak/(k+1)$ objects distribute in such a nice way fulfilling $\forall (i,j) \in \{2,...,k\}^2 : |w_x(B_i) - w_x(B_j)| \leq w_k$.

At this point of time the volume of each bin according to the pigeonhole-principle is at least

$$\begin{aligned} w_g + \left\lfloor \frac{Ak/(k+1)}{k-1} \right\rfloor &\geq w_g + \left(A\cdot\frac{k}{k^2-1} - 1\right) \\ &\overset{(*)}{>} w_g + \frac{A}{k+1}\cdot w_k. \end{aligned}$$

Since reaching x' took only $O(n)$ generations, one can show with coupon-collector's-arguments (see [10]) that $A = \omega(1)$ holds with probability $1-o(1)$. A simpler argumentation will also do the trick. Imagine n different cards. Draw each time uniformly at random one card and put it back. From former analyses (e. g., [3]) we know that having seen all available n cards takes an expected number of $\Omega(n \log n)$ attempts. The situation here is quite similar. The number of cards *not* seen after $O(n)$ attempts corresponds to A, since "seeing a card" corresponds to "move an object out of B_1". If A were a constant, $O(n)$ attempts/generations would be sufficient to see all cards: after $O(n)$ attempts only a constant number of unseen cards is left; after an expected number of $O(n)$ attempts a particular card is drawn. Therefore, an expected number of $O(n)$ attempts are enough to see all cards. This is a contradiction to the stated $\Omega(n \log n)$ attempts, hence, $A = \omega(1)$ holds – even with probability $1-o(1)$.

With this information one can show the correctness of $Ak/(k^2-1) - 1 > A/(k+1)$ and therefore the inequality $(*)$. Due to the fact that the volume of B_1 is bounded above by $2\cdot w_g + A/(k+1)\cdot w_k$, the difference between B_1 and each bin B_2 to B_k is less than w_g and condition (C2) fulfilled. Therefore, the RLS reaches the desired search point x^* with at least constant probability and the claim follows. □

A similar, but much less significant result can be shown for the (1+1)EA on g and W_ε. It states that the expected number of generations the (1+1)EA needs to find a $(2k/(k+1)-\varepsilon)$-approximation on W_ε is bounded below exponentially. But the probability of an exponential running time can only be bounded below by $2^{-\Omega(n)}$ at the moment, hence, it maybe occurs "almost never".

$O(\log w)$ bits are sufficient to encode a natural number w. Hence, it is possible to encode a number of exponential size by using only a polynomial number of bits. The bound for the approximation time in Theorem 2 is only polynomial if the sizes of the object volumes are polynomial, due to the fact that it takes into account not only the number of objects but also the volume of the objects. Therefore, it is pseudo-polynomial (see [5]) and we want to improve it. At the moment, we only succeed in doing this for $k=3$. In order to do this, we will introduce another fitness function h: it measures the difference between the volume of the fullest and the volume of the emptiest bin.

THEOREM 4. *Given $\varepsilon > 0$ RLS working on h with $k = 3$ finds a $(7+\varepsilon)$-approximation with respect to f within $O(n^2 \log n)$ generations in expectation.*

PROOF. In our previous proofs we saw that moving objects from the currently fullest into the currently emptiest bin means reducing the fitness value, until a certain approximation ratio is reached. The main idea now is to show that each object can only be transferred $O(\log n)$ times in this way until the required approximation ratio is obtained. Assuming each object can only be transferred $O(\log n)$ times from the fullest into the emptiest bin and using the fact that an appropriate transfer has the probability of at least $1/(kn)$, an expected number of $O(n^2 \log n)$ generations are sufficient for RLS to execute the at most $O(n \log n)$ transfers. The only remaining thing to show is that each object can only be transferred $O(\log n)$ times in the mentioned way.

In the following, we enumerate the bins according to their volumes in non-increasing order with respect to any arbitrary search point y, i. e., $w_y(B_y^1) \geq w_y(B_y^2) \geq w_y(B_y^3)$. We denote the current search point in generation t by x_t. Let us focus on a particular object i transferred from the fullest bin $B_{x_t}^1$ into the emptiest bin $B_{x_t}^3$ at time t. Imagine this object will be transferred again from $B_{x_{t'}}^1$ into $B_{x_{t'}}^3$ at some time $t' > t$. Now we will justify that the discrepancy between the currently fullest and the currently emptiest bin – and therefore the fitness value – decreased by a constant factor going from t to t'. Assuming that it decreases each time a object is transferred in this way by the factor $1+\delta$, it will be below $(6+\varepsilon)\cdot w_1$ after $\log_{1+\delta}(n/(6+\varepsilon))$ times, since $w \leq n \cdot w_1$ holds. Due to the fact that the volume of the emptiest bin is at most the optimal f-value, this is a search point giving the desired approximation ratio.

Have a look at object i in $B_{x_t}^1$, assuming it will be transferred into $B_{x_t}^3$ within generation t. Note, that by using h objects can only be transferred into emptier bins. Hence, in order to get i back into the fullest bin at some time $t' > t$ again, i. e., into $B^1_{x_{t'}}$, $B^3_{x_t}$ has to be filled up to become $B^1_{x_{t'}}$.

By transferring objects from bin B into B', these bins only can "balance" their volumes. Starting with a volume of v in B and v' in B', as B' gets the fuller of both bins, its volume is bounded from above by $(v + v')/2 + w_1$. Since we are interested in decreasing the volume of $B^1_{x_t}$, we have a look at the worst-case, first "balancing" $B^2_{x_t}$ and $B^3_{x_t}$ and then $B^1_{x_t}$ and $B^2_{x_t}$. We are only interested in the discrepancy between these bins and assume w. l. o. g. $B^3_{x_t}$ to be empty, while $B^1_{x_t}$ contains the volume v and $B^2_{x_t}$ v'. Now, according to our previous argumentation, the volume of $B^1_{x_{t'}}$ (after the "exchanges") is limited by

$$\frac{v + (v' + 0)/2 + w_1}{2} + w_1 \le \frac{v}{2} + \frac{v}{4} + \frac{w_1}{2} + w_1 = \frac{3}{4} \cdot v + \frac{3}{2} \cdot w_1.$$

The volume of the fullest bin is at most v greater than the volume of the lowest bin – remember we are interested in discrepancies of the volumes. The volume of the lowest bin is at any time at most as big as the optimal makespan. Therefore, the requested approximation ratio would already be reached if $v \le (6 + \varepsilon) \cdot w_1$ holds. Hence, assuming $v > (6 + \varepsilon) \cdot w_1$, we can estimate this by

$$\frac{3}{4} \cdot v + \frac{3}{2} \cdot \frac{v}{6 + \varepsilon} = \frac{24 + 3\varepsilon}{24 + 4\varepsilon} \cdot v = \frac{v}{1 + \delta}.$$

Thus we see, that the discrepancy between $B^1_{x_t}$ and $B^3_{x_t}$ has diminished by a constant factor $1 + \delta$ compared to the discrepancy between $B^1_{x_{t'}}$ and $B^3_{x_{t'}}$. This is rounding off the proof. $\square$

4. AVERAGE-CASE ANALYSIS

Theorems 1, 2 and 4 hold for an arbitrary instance for the k-Partition-Problem. But, if we specify the class of input instances, we can improve our results for the (1+1)EA on them. The class of inputs for the k-Partition-Problem we will look at can easily be described: the volumes for the n objects are drawn independently and uniformly at random from the interval $[0, 1]$. Frenk and Rinnooy Kan [4] were able to prove that the approximation ratio of the LPT rule – greedily putting the objects in non-increasing order into the currently emptiest bin – is in the magnitude of $O(\log n/n)$ on this input model and, therefore, converges to optimality as n grows large. Witt showed in [15] nearly the same result for the (1+1)EA and $k = 2$. He stated that after a polynomial number of generations the discrepancy between the two bins is in the magnitude of $O(\log n/n)$ with probability $1 - O(1/n^c)$ for any constant $c \ge 1$.

Therefore, we will take a look at the behavior of the (1+1)EA on the k-Partition-Problem for $k = 3$.

Lemma 4. *Let $n \in \mathbb{N}$ and h be defined as above for an instance of the 3-Partition-Problem. After $O(wn \log n/w_n)$ generations both introduced RSHs working on h will find a search point x fulfilling $h(x) \le 2 \cdot w_1$ with probability at least $1 - O(1/n^c)$ for an arbitrary constant $c \ge 1$.*

Proof. We will use the fitness levels $L_i := \{x | w - i \cdot w_n < h(x) \le w - (i - 1) \cdot w_n\}$ and enumerate the bins in the following by their volumes in non-increasing order, i. e., $w_x(B^1_x) \ge w_x(B^2_x) \ge w_x(B^3_x)$. Assuming $w_x(B^1_x) - w_x(B^3_x) = h(x) > 2w_1$ for the current search point x, the discrepancy $w_x(B^1_x) - w_x(B^2_x)$ or $w_x(B^2_x) - w_x(B^3_x)$ exceeds w_1. Therefore, an object transferred from B^1_x to B^3_x reduces the h-value at least by its volume. Hence, the RSH leaves the current fitness level. Since at most w/w_n fitness levels have to be left and the probabilities of the particular 1-bit-mutations sum up to at least $1/(ekn)$, the levels have been left after an expected number of $w/w_n \cdot ekn$ generations.

Let A be the event that the requested approximation ratio is obtained and let Λ denote the actual number of fitness levels to be left, i. e., $1 \le A \le w/w_n$. Define X_i as the indicator of the event that an above mentioned 1-bit-mutation occurs within the i-th generation, i. e., $X_i = 1$ iff such a 1-bit-mutation occurs. Hence, $\text{Prob}(X_i = 1) \ge 1/(ekn)$ holds and by defining $G := 8eck(\log n)w/w_n$ and $X := \sum_{i=1}^{G} X_i$ we get

$$\begin{aligned} \text{Prob}(\Lambda) &\ge \text{Prob}(X \ge A) \\ &\ge \text{Prob}(X \ge (1 - 1/2) \cdot 8c \cdot A \cdot \log n \cdot w/w_n). \end{aligned}$$

Since the mutations done in Step 2 of the introduced RSH are independent, $\text{E}(X) \ge 8c(\log n)w/w_n$ holds for the mean-value and therefore applying Chernoff bounds (e. g., [10]) reveals

$$\text{Prob}(\Lambda) \ge 1 - \text{Prob}(X < (1 - 1/2) \cdot \text{E}(X) > 1 - e^{-c \cdot A \cdot \log n}$$

and the claim follows. $\square$

The following technical lemma will become useful soon.

Lemma 5. *Given a subset $C \subsetneq \{1, ..., k\}$ of all bins, there either exists a pair $(i, i + 1)$, where object i is in a bin of C and $i + 1$ is not, or object n is in a bin of C.*

Proof. We will present a constructive proof for our claim. Have a look at the following algorithm.

- Start with the biggest object in a bin of C, namely i.
- If $i + 1$ is in a bin of $\bar{C}$, return the pair $(i, i + 1)$ else repeat with $i + 1$ instead of i until $i = n$.

Obviously, this algorithm finds such a pair or object n is in a bin of C. $\square$

Theorem 5. *Let $n \in \mathbb{N}$ and h be defined as above, while the volumes $w_1, ..., w_n$ are drawn uniformly at random from $[0, 1]$. After $O(n^{c+4} \log n)$ generations of the (1+1)EA the discrepancy between the fullest and the emptiest bin is bounded above by $O(\log n/n)$ with probability $1 - O(1/n^c)$ for an arbitrary constant $c \ge 1$. Moreover, the expected h-value after $O(n^5 \log n)$ generations is bounded above by $O(\log n/n)$.*

Proof. First, we are going to show that the discrepancy between the fullest and emptiest bin, i. e., the fitness value, is bounded above by 2 after $O(n^3 \log n)$ steps with probability $1 - O(1/n^c)$. To achieve this we use Lemma 4. In order to be able to apply Lemma 4 we need an upper bound on w and a lower bound on w_n, respectively. Obviously, $w \le n$ holds. The volumes of the objects are now drawn at random. To emphasize this, we use capitals for their notation and define W_i as the random variable describing the volume of the i-th largest object. Theory concerning order-statistics (e. g., [2]) assures $\text{Prob}(W_i - W_{i+1} \ge t) = \text{Prob}(W_n \ge t) = (1 - t)^n$ is correct. All further assumptions about the volumes we are going to make, hold at least with probability $1 - O(1/n^{c+1})$. Since we are going to make $O(n)$ independent assumptions, they conjointly hold with probability $1 - O(1/n^c)$ as required. By defining $t^* := n^{-c-2}$ we see that $W_n \ge t^* =$

$\Omega(n^{-c-2})$ and $W_i - W_{i+1} \geq t^* = \Omega(n^{-c-2})$ is correct with appropriate probability. Now, Lemma 4 implies a discrepancy of at most 2 after $O(n^3 \log n)$ generations.

Assuming its espousal, we have to show that the discrepancy drops to $O(\log n/n)$ within $O(n^{c+4} \log n)$ generations with probability $1 - O(1/n^c)$. To be able to use methods introduced in the previous sections, we would have to guarantee the existence of objects with a volume of at most $O(\log n/n)$ within the fullest bin. This could happen to be very hard, since $W_1 = \Omega(1)$ is fulfilled with high probability. But exchanging two objects i and j between two bins, can be interpreted as transferring an object with volume $W_i - W_j$. Therefore, it could be sufficient to bound these discrepancies from above by $O(\log n/n)$. Since the (1+1)EA is able to perform 2-bit-mutations with "good" probability, we take a closer look at these exchanges.

We enumerate the bins according to their volumes non-increasingly ordered. By defining $l^* := (c+1)\cdot(\ln n)/n$, $W_i - W_{i+1} \leq l^*$ and $W_n \leq l^*$ are correct with a sufficiently high probability. As long as $h(x) > 2l^*$ holds for the current search point x, the discrepancy between B_x^1 and B_x^2 or B_x^2 and B_x^3 exceeds l^* due to the pigeonhole-principle. We will take care of the first mentioned case first. By defining $C := \{1\}$ Lemma 5 provides us with a pair $(i, i+1)$ of objects or object n is within bin B_x^1. A transfer of n into bin B_x^3 or an exchange of i and $i+1$, respectively, will reduce the volume of B_x^1 by at least $W_n \geq t^* = \Omega(n^{-c-2})$ or $W_i - W_{i+1} \geq t^* = \Omega(n^{-c-2})$, respectively. As our assumptions on the volumes hold, it will not be diminished by more than l^*. Since the discrepancy between B_x^1 and B_x^2 is greater than l^*, the h-value reduces at least by $t^* = \Omega(n^{-c-2})$.

On the other hand, if the second case occurs, i. e., the discrepancy of B_x^2 and B_x^3 exceeds l^*, the same method can be used, redefining $C := \{1, 2\}$, due to the fact that the volume of B_x^3 increases by at least $t^* = \Omega(n^{-c-2})$ and at most l^*.

Therefore, the (1+1)EA is able to reduce the h-value by at least $t^* = \Omega(n^{-c-2})$ by transferring at most two objects, i. e., performing a particular at most 2-bit-mutation. The probability for this to happen is bounded below by $(kn)^{-2}$. Hence, after at most $2/t^* = O(n^{c+2})$ – remember we started with a discrepancy of at most 2 – reductions the (1+1)EA has found a appropriate search point. Due to Chernoff bounds, this is going to happen after $O(n^{4+c} \log n)$ generations with probability $1 - O(1/n^c)$. This gives the first claim.

To prove the claim regarding the expected discrepancy we substitute $c = 2$ in Lemma 4. The discrepancy between B_x^1 and B_x^3 is trivially bounded above by n and with a probability of at most $O(n^{-2})$ it is not bounded above by 2 after $O(n^5 \log n)$ generations. This will yield a term of $O(1/n)$ to the expected discrepancy in that case. By using $c = 1$ in argumentation above, we see that with probability at most $O(1/n)$ the discrepancy after $O(n^5 \log n)$ generations is bounded above by at most 2; again yielding a term of $O(1/n)$ to the expected discrepancy. This finally proves our claim. □

5. CONCLUSIONS

Although evolutionary algorithms are non-specialized, general purpose, randomized search heuristics, they are able to deliver appropriate approximative solutions in acceptable running time. This holds for "simple" problems (see [3] and [6]) as well as for some sort of NP-hard problems, like the k-Partition-Problem. They sometimes can even compete with problem-specialized algorithms, e. g., the LPT rule. We presented an analysis proving an expected pseudo-polynomial running time to create a $(2k/(k+1))$-approximation on this problem using the (1+1)EA and RLS. The usage of a different proof method reveals an $O(n^2 \log n)$ bound for a $(7 + \varepsilon)$-approximation on the 3-Partition-Problem. On a common input distribution the (1+1)EA is even capable of finding a solution for the 3-Partition-Problem with a discrepancy of at most $O(\log n/n)$, i. e., convergent to optimality, in polynomial time with high probability. Nevertheless, the $(1+1)$EA and RLS take exponential expected running time to find a $(2k/(k+1) - \varepsilon)$-approximation in the worst-case for a constant $\varepsilon > 0$.

Hopefully, we did a step toward the analysis of the average-case behavior of evolutionary algorithms on important problems.

6. REFERENCES

[1] T. Baeck, D. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation.* Institute of Physics Publishing, 1997.

[2] H. David and H. Nagaraja. *Order Statistics, 3rd edition.* Wiley, 2003.

[3] S. Droste, T. Jansen, and I. Wegener. On the Analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science*, 276:51–81, 2002.

[4] J. Frenk and A. Rinnooy Kan. The Rate of Convergence to Optimality of the LPT Rule. In *Discrete Applied Mathematics*, volume 14, pages 187–197. 1986.

[5] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[6] O. Giel and I. Wegener. Evolutionary Algorithms and the Maximum Matching Problem. In *Proceedings of 20th STACS*, number 2607 in LNCS, pages 415–426, 2003.

[7] R. Graham. Bounds for Certain Multiprocessing Anomalies. In *Bell System Technical Journal*, number 45, pages 1563–1581. 1966.

[8] D. Hochbaum and D. Shmoys. Using Dual Approximation Algorithms for Scheduling Problems: Theoretical and Practical Results. In *Journal of the Association for Computing Machinery*, number 34 (1), pages 144–162. 1987.

[9] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by Simulated Annealing. In *Science*, volume 220, 4598, pages 671–680. 1983.

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

[11] F. Neumann and I. Wegener. Randomized Local Search, Evolutionary Algorithms, and the Minimum Spanning Tree Problem. In *Proc. of GECCO, LNCS*, volume 3102, pages 713–724. 2004.

[12] S. Sahni. Algorithms for Scheduling Independent Tasks. In *Journal of the Association for Computing Machinery*, number 23 (1), pages 116–127. 1976.

[13] V. Vazirani. *Approximation Algorithms, Corrected Second Printing.* Springer, 2003.

[14] I. Wegener. Methods for the Analysis of Evolutionary Algorithms on Pseudo-Boolean Functions. In R. Sarker, X. Yao, M. Mohammadian, editor, *Evolutionary Optimization*, pages 349–369. Kluwer, 2002.

[15] C. Witt. Worst-Case and Average-Case Approximations by Simple Randomized Search Heuristics. In *Proceedings of STACS*, volume 3404 of *LNCS*, pages 44–56, 2005.