UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

On the Analysis of a Simple Evolutionary Algorithm With Dynamic Parameter Control

Stefan Droste and Thomas Jansen

No. CI-68/99

Technical Report ISSN 1433-3325 April 1999

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence", at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

On the Analysis of a Simple Evolutionary Algorithm With Dynamic Parameter Control*

Stefan Droste Thomas Jansen
FB Informatik, LS 2, Univ. Dortmund, D-44221 Dortmund, Germany
{droste, jansen}@ls2.cs.uni-dortmund.de

Abstract

Evolutionary algorithms usually are controlled by various parameters and it is well known that an appropriate choice of these control parameters is crucial for the efficiency of the algorithms. In many cases it seems to be favorable not to use a static set of parameter settings for a problem but let the sizes of the parameters vary during an optimization. Even for the most simple type of non-static parameter settings, dynamic parameter control, no formal general proof is known that varying the parameter settings is advantageous. Here, a very simple evolutionary algorithm is analyzed, and an exponential improvement against even the optimal static parameter setting is proved. This result is closely related to the open question whether simulated annealing with a natural cooling schedule can provably outperform the Metropolis algorithm on a natural problem.

1 Introduction

When designing optimization algorithms, it seems much more natural to use algorithms that adapt their search strategy accordingly to the information they gather during a run. We consider evolutionary algorithms (EAs), which are general search heuristics that can be used for static function optimization. Evolutionary algorithms use strong abstractions of the principles of natural evolution. While many other optimization algorithms use only one single element of the search space, EAs typically employ a whole *population* of them. They generate new search points using genetic operators like *recombination* or *mutation* and *select* the next population out of these and the old population. This is done until some stopping criterion is fulfilled. Depending on the representation, the genetic operators, and the selection method used the resulting EA belongs to the paradigm of genetic algorithms (see Holland (1975)), evolution strategies (see Schwefel (1995)), evolutionary programming (see Fogel (1995)) or genetic programming (see Koza (1992)). Although there are many successful applications of EAs, theoretically founded knowledge is in general rare.

*This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (531).

Evolutionary algorithms are controlled by a number of different parameters, and it is well known, that an appropriate choice of the parameter settings is crucial for an efficient optimization. In the simplest case the parameter settings are kept constant during a run, but it is at least intuitively clear, that varying the parameter settings during a run can be advantageous. Bäck (1998) divides the class of evolutionary algorithms with non-static parameter settings into three subclasses. The simplest model is that of *dynamic parameter control*, where the parameters settings are modified according to some fixed schedule. A more sophisticated model is *adaptive parameter control*, where the new choice of the parameters is guided by the success during the optimization so far. Finally, using *self-adaptive parameter control* means that the parameter values are evolved in a way similar to the treatment of the population of the EA, which is evolved by the application of randomized evolutionary variation operators and selection.

Though all three models are successfully applied, even for the most simple variant, the dynamic parameter control, there exists only little formal knowledge. Because of this lack of a theoretical basis, the analysis of most EAs used in practice is too difficult. To lay the foundations it is therefore advisable to start with simple EAs. One of them is the so-called (1+1) EA (see Rudolph (1997)) used for maximization of an objective function $f : \{0,1\}^n \rightarrow \mathbb{R}$. Its population consists of only one bit string of length n, which is randomly initialized in the beginning. To generate a new search point, every bit of the old bit string is flipped with probability $1/n$, which is selected for the new population, if it has at least the same objective function value as the old one. Otherwise, the element of the old population forms the new one. The expected running times of the (1+1) EA for different classes of objective functions are analyzed theoretically in Rudolph (1997), Droste, Jansen, and Wegener (1998b,1998c).

Here we present an analysis of a similar and also very simple evolutionary algorithm, where we prove for two examples that there is an exponential gap in the expected running time between an appropriate dynamic parameter control and the optimal static choice for the parameter settings.

In the following we analyze a variant of the (1+1) EA, where mutation consists of flipping exactly one randomly chosen bit. While this makes an analysis much easier, the selection is now more complicated: if the new search point is x' and the old one x, the new point x' is selected with probability $\min(1, \alpha^{f(x')-f(x)})$, where the *selection parameter* α is an element of $[1, \infty[$. So worsenings are now accepted with some probability, which decreases for large worsenings, while improvements are always accepted.

The only parameter for which we consider static and non-static settings is the selection parameter α. For two concrete functions to be maximized, we show that the expected running time until the maximum is found for the first time is polynomial when dynamic parameter control is employed appropriately, but exponential for static α regardless of the chosen α. We do this by showing, that the running times with dynamic parameter control are exponential only with exponentially small probability, while they can be polynomially upper bounded in all the other cases. This implies the results about the expected running times.

To avoid any misunderstandings we present the algorithms, which are for sake of brevity denoted by DYNAMIC EA and STATIC EA, more formally now.

Algorithm 1.1: STATIC EA

1. Choose $x \in \{0, 1\}$ uniformly at random.
2. Create y by flipping one uniformly chosen random bit from x.
3. With probability $\min\{1, \alpha^{f(y)-f(x)}\}$ set $x := y$.
4. Continue at 2.

Algorithm 1.2: DYNAMIC EA

1. Choose $x \in \{0, 1\}$ uniformly at random. Set $t := 1$.
2. Create y by flipping one uniformly chosen random bit from x.
3. With probability $\min\{1, \alpha(t)^{f(y)-f(x)}\}$ set $x := y$.
4. Set $t := t + 1$. Continue at 2.

The function $\alpha : \mathbb{N} \to [1; \infty[$ is usually denoted as *selection schedule*. We note that STATIC EA is an instance of the Metropolis algorithm (see Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953)), while DYNAMIC EA is a simulated annealing algorithm, where the neighborhood of a search point consists of all points with Hamming distance one. Hence, our approach can be seen also as a step to answer the question raised by Jerrum and Sinclair (1997): Is there a natural cooling schedule (which corresponds to our selection schedule), so that simulated annealing provenly outperforms the Metropolis algorithm for a natural problem? There are various attempts to answer this question. We mention Jerrum and Sorkin (1998) which prove that already the Metropolis algorithm can efficiently solve the problem of graph bisection on a special type of random graphs. Another valuable attempt is presented by Sorkin (1991), who proves that simulated annealing, i. e. dynamic parameter control, is superior to the Metropolis algorithm, i. e. static choice of α, on a carefully designed fractal function. He proves his results using the method of rapidly mixing Markov chains (see Sinclair (1993) for an introduction). Note, that our examples have a much simpler structure and are easier to understand. Furthermore, we derive our results using quite elementary methods. Namely, our proofs do chiefly use Chernoff bounds (Hagerup and Rüb 1989).

We use the expected number of steps the algorithms need to reach a global maximum of the objective function f for the first time to measure the performance of the algorithms. The analysis of DYNAMIC EA and STATIC EA is done for two special, artificial functions, namely MODJUMP_2 and VALLEY. Both functions are symmetric, i. e. their function value depends only on the number of ones in its argument. Hence, we will present in the next section some basic properties of STATIC EA for symmetric functions. For MODJUMP_2

we show in Section 3 that for a class of selection schedules, where the probability of accepting worsenings is rising during the run of the algorithm, DYNAMIC EA outperforms STATIC EA for any constant choice of α. The gap between the expected running times is proven to be exponential. For VALLEY we show in Section 4 the same for a class of selection schedules, where the probability of accepting worsenings decreases during the run. Finally, we draw some conclusions and make some remarks about possible future research.

2 Basic Properties of STATIC EA for symmetric functions

In this section we derive some equations for the expected number of steps, STATIC EA needs to find a maximum. If we can bound the value of $\alpha(t)$, which DYNAMIC EA uses, these equations will be also helpful to bound the expected number of steps of DYNAMIC EA. So even if we speak only of STATIC EA in the following, all results can be used for DYNAMIC EA, too, if we have a bound for $\alpha(t)$.

We assume that our objective functions are symmetric and have their only global maximum at the all ones bit string $(1, \ldots, 1)$. These assumptions are valid for the functions MODJUMP_2 and VALLEY we analyze in Sections 3 and 4. A symmetric function is defined as follows:

Definition 2.1: A function $f : \{0,1\}^n \to \mathbb{N}$ is called *symmetric*, if there is a vector $(f_0, f_1, \ldots, f_n) \in \mathbb{N}^{n+1}$, so that

$$\forall x = (x_1, \ldots, x_n) \in \{0,1\}^n : \left(\sum_{i=1}^{n} x_i = j \Rightarrow f(x) = f_j \right).$$

So, when trying to maximize a symmetric function with STATIC EA, the expected number of steps the algorithm needs depends only on the number of ones the actual bit string x contains, but not on their position. Therefore, it can be modeled by a Markov chain with exactly $n+1$ states. Let the random variable T_i (for $i \in \{0, \ldots, n\}$) be the random number of steps STATIC EA needs to reach the maximum for the first time, when starting in a bit string with i ones. As the initial bit string is chosen randomly with equal probability, the expected value of the number T of steps, the whole algorithm needs, is

$$\mathrm{E}(T) = \sum_{i=0}^{n} \frac{\binom{n}{i}}{2^n} \cdot \mathrm{E}(T_i).$$

So, by bounding $E(T_i)$ for all $i \in \{0, \ldots, n\}$ we can bound $E(T)$. As STATIC EA can only change the number of ones in its actual bit string by one, the number T_i of steps to reach the maximum $(1, \ldots, 1)$ is the sum of the numbers T_j^+ of steps to reach $j+1$ ones, when starting with j ones, over all $j \in \{i, \ldots, n-1\}$:

$$T_i = T_i^+ + T_{i+1}^+ + \ldots + T_{n-1}^+.$$

Let p_i^+ resp. p_i^- be the transition probability, that STATIC EA goes to a state with $i+1$ resp. $i-1$ ones when being in a state with $i \in \{0, \dots, n\}$ ones. Then it directly follows:

Lemma 2.2: *The expected number $E(T_i^+)$ of steps to reach a state with $i+1$ ones for the first time, when starting in a state with $i \in \{1, \dots, n-1\}$ ones, is:*

$$E(T_i^+) = \frac{1}{p_i^+} + \frac{p_i^-}{p_i^+} \cdot E(T_{i-1}^+).$$

Proof: When being in a state with $i \in \{1, \dots, n-1\}$ ones, the number of ones can increase, decrease or stay the same. This leads to the following equation:

$$\begin{aligned}
& E(T_i^+) && = p_i^+ \cdot 1 + p_i^- \cdot \left(1 + E(T_{i-1}^+) + E(T_i^+)\right) + \left(1 - p_i^+ - p_i^-\right) \cdot \left(1 + E(T_i^+)\right) \\
\Leftrightarrow \quad & p_i^+ \cdot E(T_i^+) && = 1 + p_i^- \cdot E(T_{i-1}^+) \\
\Leftrightarrow \quad & E(T_i^+) && = \frac{1}{p_i^+} + \frac{p_i^-}{p_i^+} \cdot E(T_{i-1}^+).
\end{aligned}$$

□

Using this recursive equation to determine $E(T_i^+)$, we can derive the following lemma:

Lemma 2.3: *The expected number $E(T_i^+)$ of steps to reach a state with $i+1$ ones for the first time, when starting in a state with $i \in \{1, \dots, n-1\}$ ones, is for all $j \in \{1, \dots, i\}$:*

$$E(T_i^+) = \left(\sum_{k=0}^{j-1} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{j-1} p_{i-l}^-}{\prod_{l=0}^{j-1} p_{i-l}^+} \cdot E(T_{i-j}^+).$$

Proof: The equation can be proven by induction over j. For $j = 1$ it is just Lemma 2.2. Assuming that it is valid for j, we can prove it for $j+1$ in the following way:

$$\begin{aligned}
E(T_i^+) &= \left(\sum_{k=0}^{j-1} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{j-1} p_{i-l}^-}{\prod_{l=0}^{j-1} p_{i-l}^+} \cdot E(T_{i-j}^+) \\
&= \left(\sum_{k=0}^{j-1} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{j-1} p_{i-l}^-}{\prod_{l=0}^{j-1} p_{i-l}^+} \cdot \left(\frac{1}{p_{i-j}^+} + \frac{p_{i-j}^-}{p_{i-j}^+} \cdot E(T_{i-j-1}^+) \right) \\
&= \left(\sum_{k=0}^{j-1} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{j-1} p_{i-l}^-}{\prod_{l=0}^{j} p_{i-l}^+} + \frac{\prod_{l=0}^{j} p_{i-l}^-}{\prod_{l=0}^{j} p_{i-l}^+} \cdot E(T_{i-j-1}^+) \\
&= \left(\sum_{k=0}^{j} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{j} p_{i-l}^-}{\prod_{l=0}^{j} p_{i-l}^+} \cdot E(T_{i-(j+1)}^+).
\end{aligned}$$

□

Because $E(T_0^+) = 1/p_0^+$, we get for the case $j = i$:

Corollary 2.4: *The expected number $E(T_i^+)$ of steps to reach a state with $i+1$ ones, when starting in a state with $i \in \{1, \ldots, n-1\}$ ones, is:*

$$E(T_i^+) = \left(\sum_{k=0}^{i-1} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} \right) + \frac{\prod_{l=0}^{i-1} p_{i-l}^-}{\prod_{l=0}^{i-1} p_{i-l}^+} \cdot \frac{1}{p_0^+} = \sum_{k=0}^{i} \frac{\prod_{l=0}^{k-1} p_{i-l}^-}{\prod_{l=0}^{k} p_{i-l}^+} = \sum_{k=0}^{i} \frac{1}{p_k^+} \cdot \prod_{l=k+1}^{i} \frac{p_l^-}{p_l^+}.$$

3 The function ModJump$_2$

We consider the function ModJump$_2$ that is a modification of the function Jump$_2$ introduced by Droste, Jansen, and Wegener (1998a). Though ModJump$_2$ is in some sense a fairly well structured and therefore easy to analyze function, it turns out that it is helpful to take a look at an even simpler function, namely OneMax, before.

Definition 3.1: The function OneMax : $\{0,1\}^n \to \mathbb{N}$ is defined by

$$\text{OneMax}(x) := ||x||_1,$$

where $||x||_1 = x_1 + \cdots + x_n$ denotes the number of ones in x.
The function ModJump$_2$: $\{0,1\}^n \to \mathbb{N}$ is defined by

$$\text{ModJump}_2(x) := \begin{cases} ||x||_1 & \text{for } ||x||_1 \le n-3. \\ 3n^2 + n & \text{for } ||x||_1 = n-2. \\ 3n^2 & \text{for } ||x||_1 = n-1. \\ 3n^2 + n + 1 & \text{for } ||x||_1 = n. \end{cases}$$

Since OneMax is a symmetric function we can use the results of Section 2. They lead us directly to:

Lemma 3.2: *The expected number of steps until* Static EA *reaches for the first time the maximum of* OneMax *when started in a bit string with exactly $i < n$ ones equals*

$$\mathrm{E}(T_i) = \sum_{j=i}^{n-1} \mathrm{E}\left(T_j^+\right) = \sum_{j=i}^{n-1} \left(\frac{1}{\alpha^j \binom{n-1}{j}} \sum_{k=0}^{j} \binom{n}{k} \alpha^k \right),$$

where

$$\alpha \left(\left(1 + \frac{1}{\alpha}\right)^n - 1 \right) \le \mathrm{E}(T_i) \le \left(1 + \frac{1}{\alpha}\right)^{n-1} \cdot n \left(\ln(n-i) + 1\right).$$

Proof: It is easy to recognize that for all $i \in \{0, \ldots, n-1\}$:

$$p_i^+ = \frac{n-i}{n} \text{ and } p_i^- = \frac{i}{n} \cdot \alpha^{(i-1)-i} = \frac{i}{\alpha n}$$

holds for OneMax. By Corollary 2.4 we get

$$\begin{aligned}
\mathrm{E}\left(T_j^+\right) &= \sum_{k=0}^{j} \frac{n}{n-k} \prod_{l=k+1}^{j} \frac{l}{\alpha(n-l)} = \sum_{k=0}^{j} \frac{n}{n-k} \cdot \frac{j! \cdot (n-j-1)!}{\alpha^{j-k} \cdot k! \cdot (n-k-1)!} \\
&= \frac{j! \cdot (n-j-1)!}{\alpha^j \cdot (n-1)!} \sum_{k=0}^{j} \frac{n!}{n-k} \cdot \frac{\alpha^k}{k! \cdot (n-k-1)!} = \frac{1}{\alpha^j \binom{n-1}{j}} \sum_{k=0}^{j} \binom{n}{k} \alpha^k,
\end{aligned}$$

which proves the exact formulation for $\mathrm{E}(T_i)$.
For the upper bound we consider $\mathrm{E}\left(T_j^+\right)$ and see that

$$\begin{aligned}
\mathrm{E}\left(T_j^+\right) &= \frac{1}{\alpha^j \binom{n-1}{j}} \sum_{k=0}^{j} \binom{n}{k} \alpha^k = \frac{j!(n-j-1)!}{\alpha^j (n-1)!} \sum_{k=0}^{j} \frac{n! \alpha^k}{k!(n-k)!} \\
&= \frac{j!(n-j-1)!}{\alpha^j (n-1)!} \sum_{k=0}^{j} \frac{n! \alpha^{j-k}}{(j-k)!(n-j+k)!} \\
&= \frac{n}{n-j} \sum_{k=0}^{j} \alpha^{-k} \frac{\binom{j}{k}}{\binom{n-j+k}{n-j}} \leq \frac{n}{n-j} \sum_{k=0}^{j} \binom{j}{k} \left(\frac{1}{\alpha}\right)^k = \frac{n}{n-j} \left(1 + \frac{1}{\alpha}\right)^j
\end{aligned}$$

holds. So we have

$$\begin{aligned}
\mathrm{E}(T_i) &= \sum_{j=i}^{n-1} \mathrm{E}\left(T_j^+\right) \leq \sum_{j=i}^{n-1} \frac{n}{n-j} \left(1 + \frac{1}{\alpha}\right)^j \\
&= n \left(1 + \frac{1}{\alpha}\right)^n \cdot \sum_{j=1}^{n-i} \frac{1}{j} \left(1 + \frac{1}{\alpha}\right)^{-j} \leq \left(1 + \frac{1}{\alpha}\right)^{n-1} \cdot n \left(\ln(n-i) + 1\right)
\end{aligned}$$

as upper bound.
For the lower bound we have

$$\begin{aligned}
\mathrm{E}(T_i) &\geq \mathrm{E}\left(T_{n-1}^+\right) = \frac{1}{\alpha^{n-1} \binom{n-1}{n-1}} \sum_{k=0}^{n-1} \binom{n}{k} \alpha^k = \frac{\left(\sum_{k=0}^{n} \binom{n}{k} \alpha^k\right) - \alpha^n}{\alpha^{n-1}} \\
&= \frac{(1+\alpha)^n - \alpha^n}{\alpha^{n-1}} = \alpha \left(\left(1 + \frac{1}{\alpha}\right)^n - 1\right).
\end{aligned}$$

□

Using Lemma 3.2 we can prove a lower bound on the expected running time of Static EA on ModJump$_2$.

Theorem 3.3: *The expected number of steps until* Static EA *reaches the maximum of* ModJump$_2$ *for the first time is*

$$\Omega\left(n \cdot \alpha^n + \frac{\left(1 + \frac{1}{\alpha}\right)^{n-3}}{n^2}\right)$$

which is exponential for all choices of $\alpha \in [1; \infty[$.

Proof: We model the run of STATIC EA on MODJUMP$_2$ as a Markov chain as we did for ONEMAX. For all bit strings with $i \leq n-3$ ones the transition probabilities p_i^+ and p_i^- are equal for ONEMAX and MODJUMP$_2$. Therefore, for $j \leq n-3$ we still have

$$\mathrm{E}\left(T_j^+\right) = \frac{1}{\alpha^j \binom{n-1}{j}} \sum_{k=0}^{j} \binom{n}{k} \alpha^k.$$

We assume w. l. o. g. that n is even. With probability at least $1/2$ STATIC EA starts in a bit string with at most $n/2$ ones. We reconsider the derivation of the bounds in the proof of Lemma 3.2 and see that for MODJUMP$_2$

$$\begin{aligned} \mathrm{E}\left(T_{n/2}\right) &\geq \sum_{j=n/2}^{n-3} \mathrm{E}\left(T_j^+\right) \geq \mathrm{E}\left(T_{n-3}^+\right) = \frac{1}{\alpha^{n-3}\binom{n-1}{n-3}} \cdot \sum_{j=0}^{n-3} \binom{n}{j} \alpha^j \\ &\geq \frac{1}{\alpha^{n-3} n^2} \sum_{j=0}^{n-3} \binom{n-3}{j} \alpha^j = \frac{\left(1+\frac{1}{\alpha}\right)^{n-3}}{n^2} \end{aligned}$$

holds.

Additionally, we inspect the step from a bit string with $n-2$ ones to a bit string with $n-1$ ones. The probability for such a transition equals

$$\frac{2}{n} \cdot \alpha^{3n^2-(3n^2+n)} = \frac{2}{n \cdot \alpha^n}.$$

With probability $1-(n+1)/2^n$ STATIC EA starts with a bit string that contains less than $n-1$ ones. Therefore, the expected number of steps until a bit string with $n-1$ ones is reached is

$$\Omega\left(n \cdot \alpha^n\right),$$

so this lower bounds the expected running time of the algorithm. Altogether we have

$$\Omega\left(n \cdot \alpha^n + \frac{\left(1+\frac{1}{\alpha}\right)^{n-3}}{n^2}\right)$$

as lower bound for the expected number of steps as claimed. For $\alpha \geq 2$ we recognize that the bound is $\Omega\left(n \cdot 2^n\right)$ and for $\alpha < 2$ it is $\Omega\left((3/2)^n/n^2\right)$. $\square$

We see that the running time of STATIC EA is exponential regardless of the choice of α. Now we prove that an appropriate selection schedule enables DYNAMIC EA to maximize MODJUMP$_2$ in a polynomially bounded expected number of steps. In fact, our proof does not rely on one special selection schedule, but is carried our for a class of selection schedules. As mentioned above, the most important common property of these selection schedules is, that the probability of accepting worsenings is rising during the run.

Theorem 3.4: *With probability* $1 - \mathrm{O}(e^{-n})$ *the number of steps until* DYNAMIC EA *with the selection schedule*

$$\alpha(t) := \max\left\{1 + \frac{1}{n}, \frac{s(n)}{t}\right\}$$

reaches the maximum of MODJUMP$_2$ *for the first time is* $\mathrm{O}(s(n))$ *for any polynomial* $s(n)$ *with* $s(n) \geq 8en^3 \ln n$. *Furthermore, the expected number of steps until this happens is* $\mathrm{O}(s(n))$.

Proof: First of all, we notice that everything we said about STATIC EA carries over to DYNAMIC EA if we take into account that α is no longer fixed. It depends on the number of steps performed so far. Then the idea of the proof is simple. In a first phase a local maximum with $n-2$ ones is reached quite fast with high probability since $\alpha(t)$ is large enough. In a second phase with high probability the algorithm stays there long enough for $\alpha(t)$ to decrease sufficiently so that finally the maximum is reached. With small probability things go wrong. In this case we are able to find an upper bound on the running time, too. So altogether the expected running time can be polynomially upper bounded.

We distinguish two phases. The first phase ends after $s(n)/n$ steps. During the first $s(n)/n$ steps, we have $\alpha(t) \geq n$. Furthermore, since $s(n) \geq 8en^3 \ln n$, we have that $s(n)/n \geq 8en^2 \ln n$. From the proof of Lemma 3.2 it is clear, that the expected number of steps until DYNAMIC EA reaches a bit string with $n-2$ steps is bounded above by $2en \ln n$. We use Markov's inequality and see that the probability that it is not reached in $4en \ln n$ steps is bounded above by $1/2$. As our analysis is independent of the initial bit string we may in case, that a bit string with more than $n-3$ ones is not yet reached, consider the following at least $(2n-1) \cdot 4en \ln n$ steps together with the first $4en \ln n$ steps as at least $2n$ independent trials. Therefore, the probability that after the first phase the current x of DYNAMIC EA contains less then $n-2$ ones is upper bounded above by 4^{-n}. Once such a bit string is reached, the probability that a bit string with less than $n-2$ ones is reached in at most $s(n)/n$ steps is bounded above by

$$\frac{s(n)}{n}\left(\frac{n-2}{n} \cdot \alpha^{n-3-(3n^2+n)}\right) \leq n^{-3n^2+\mathrm{O}(1)}.$$

Hence, the second phase starts with a bit string having at least $n-2$ ones with high probability and is finished when the maximum of MODJUMP$_2$ is found or $s(n)+4n^3$ steps are performed (including the steps in the first phase). We assume pessimistically that the initial bit string in the second phase contains exactly $n-2$ ones. Therefore, the probability to reach a bit string with $n-3$ ones in one step is

$$\frac{n-2}{n} \cdot \alpha^{n-3-(3n^2+n)} \leq \alpha^{-3n^2}.$$

Obviously, the probability to reach a bit string with $n-3$ ones in at most $s(n)+4n^3$ steps is upper bounded by

$$\left(s(n) + 4n^3\right) \alpha^{-3n^2}.$$

We have $\alpha(t) \geq 1 + 1/n$ in all steps, so we have

$$\frac{s(n) + 4n^3}{(1+1/n)^{3n^2}} \leq \frac{n^{\mathrm{O}(1)}}{e^{3n^2/(2n)}} = e^{\mathrm{O}(\ln n) - (3/2)n} = \mathrm{O}\left(e^{-n}\right)$$

as an upper bound.

If the maximum is not reached before, there are $4n^3$ steps in the second phase where $\alpha(t) = 1 + 1/n$ holds. Then, the probability, that from a bit string with at least $n-2$ ones the maximum is reached in two steps, is lower bounded by

$$\left(\frac{2}{n} \cdot \alpha^{3n^2 - (3n^2+n)}\right) \cdot \frac{1}{n} = \frac{2}{n^2} \cdot \left(1 + \frac{1}{n}\right)^n \geq \frac{2}{en^2}.$$

The probability, that after $2n^3$ "double-steps" (i. e. $4n^3$ subsequent steps) the maximum is not reached is bounded above by

$$\left(1 - \frac{2}{en^2}\right)^{2n^3} \leq e^{-n}.$$

We combine what we have so far. The probability that the global maximum is not found after at most $s(n) + 4n^3$ steps is bounded above by

$$e^{-n} + \mathrm{O}\left(e^{-n}\right) + 4^{-n} + n^{-3n^2 + \mathrm{O}(1)} = \mathrm{O}\left(e^{-n}\right).$$

Therefore, with probability $1 - \mathrm{O}\left(e^{-n}\right)$ the maximum is reached after $\mathrm{O}\left(s(n)\right)$ steps. We consider the unlikely case that this does not happen and give an upper bound for the expected number of steps. We do not care what happens in the first $s(n) + 4n^3$ steps of the run. After that we have $\alpha(t) = 1 + 1/n$ for the rest of the time. This yields

$$p_b^+ = \frac{n-b}{n}, \quad p_b^- = \frac{b}{(1+1/n)n}$$

for $0 \leq b \leq n-3$ and

$$p_{n-2}^+ = \frac{2}{(1+1/n)^n \cdot n}, \quad p_{n-2}^- = \frac{n-2}{(1+1/n)^{3n^2+3} \cdot n}, \quad p_{n-1}^+ = \frac{1}{n}, \quad p_{n-1}^- = \frac{n-1}{(1+1/n)^n \cdot n}$$

for the rest of the cases. From Corollary 2.4 we know that the expected number of steps to come from a state with b ones to one with $b+1$ ones for the first time equals

$$\mathrm{E}\left(T_b^+\right) = \sum_{i=0}^{b} \frac{1}{p_i^+} \prod_{j=i+1}^{b} \frac{p_j^-}{p_j^+}$$

and it is obvious that this decreases with increasing values for p_i^+ and decreasing values for p_j^-/p_j^+. We compare $\mathrm{E}\left(T_b^+\right)$ to a pure random walk that corresponds to choosing $\alpha = 1$ regardless of the objective function. We see, that for $0 \leq b \leq n-3$ the probability to increase the number of ones in one step from b to $b+1$ for the pure random walk is equal to p_b^+ and the probability to decrease it is by a factor $1+1/n$ larger. For $b = n-2$ we have

$$\frac{p_{n-2}^-}{p_{n-2}^+} = \frac{n-2}{(1+1/n)^{3n^2+3}n} \cdot \frac{(1+1/n)^n n}{2} = \frac{n-2}{2(1+1/n)^{3n^2-n+3}},$$

which quickly converges to zero. For the pure random walk the corresponding value is

$$\frac{n-2}{n} \cdot \frac{n}{2} = \frac{n-2}{2}.$$

We conclude that for $0 \leq b \leq n-2$ we have that $\mathrm{E}\left(T_b^+\right)$ is upper bounded by the corresponding value for the pure random walk. The same is true for $\mathrm{E}\left(T_{n-1}^+\right)$, as can easily be seen by recognizing that the probability to increase the number of ones in one step from $n-1$ to n for the pure random walk is equal to p_{n-1}^+ and that the probability to decrease it is much larger than p_{n-1}^-. Therefore the expected number of steps a pure random walk started with b ones needs to reach the all ones string is an upper bound for $\mathrm{E}(T_b)$.

Now we investigate the expected number of steps until a pure random walk finds the all ones string for the first time. We look for an upper bound. We have to take into account that the starting point is the current x of Dynamic EA after $s(n) + 4n^3$ steps and it cannot be assumed to be random. The expected number of steps decreases with increasing number of ones b in the starting point. We have

$$\begin{aligned}
\mathrm{E}(T_i) &\leq \mathrm{E}(T_0) = \sum_{j=0}^{n-1} \frac{1}{\binom{n-1}{j}} \sum_{k=0}^{j} \binom{n}{k} < 2^n \sum_{j=0}^{n-1} \frac{1}{\binom{n-1}{j}} \\
&= 2^n \left(2 + \sum_{j=1}^{n-2} \frac{1}{\binom{n-1}{j}}\right) \leq 2^n \left(2 + \sum_{j=1}^{n-2} \frac{1}{n-1}\right) < 2^n \cdot 3,
\end{aligned}$$

so the expected number of steps is at most $3 \cdot 2^n$ for any starting point.

This yields that the contribution in case of a failure to the expected number of steps is

$$3 \cdot 2^n \cdot \mathrm{O}\left(e^{-n}\right) = \mathrm{O}(1)$$

and we have $\mathrm{O}(s(n))$ as upper bound on the expected running time as claimed. □

4 The function Valley

We showed in the previous section, that for maximizing ModJump_2 Static EA for arbitrarily chosen α needs exponential expected time, while Dynamic EA with an appropriate selection schedule $\alpha : \mathbb{N} \to [1; \infty[$ needs only polynomial expected time. The used selection schedule increased the probability of accepting worsenings with increasing t. Regarding $1/\alpha(t)$ as temperature of simulated annealing this means an increasing temperature, which is not according to the physical analogue of cooling down.

Hence, we show in this section, that there exists a function VALLEY, so that DYNAMIC EA using an appropriate selection schedule with decreasing probability for accepting worsenings needs only polynomial expected time, while STATIC EA needs exponential expected time, independent from the choice of α. Analogously to the previous section we do this by showing that the running time of DYNAMIC EA is exponential only with exponentially small probability, while it is polynomial in all other cases.

Intuitively the function VALLEY should have the following properties: for a constant fraction of points the function values, when moving to the maximum, should decrease in the beginning, so that a high probability of accepting worsenings speeds up this process. This should be followed by a rise of the function values, so that accepting worsenings would slow down the maximization. We will show that the following function fulfills these intuitive concepts to a sufficient extent:

Definition 4.1: The function $\text{VALLEY} : \{0,1\}^n \to \mathbb{N}$ is defined by (w.l.o.g. n is even):

$$\text{VALLEY} := \begin{cases} n/2 - ||x||_1 & \text{for } ||x||_1 \leq n/2 \\ (7n^2 \ln n) - n/2 + ||x||_1 & \text{for } ||x||_1 > n/2 \end{cases}.$$

Then the following holds:

Theorem 4.2: *The expected number of steps until* STATIC EA *reaches the maximum of* VALLEY *for the first time is*

$$\Omega\left(\left(\sqrt{\frac{\alpha}{4}}\right)^n + \left(\frac{1}{\alpha}+1\right)^{n-4}\right),$$

which is exponential for all choices of $\alpha \in [1;\infty[$.

Proof: When we take a look at the function VALLEY for all x with $||x||_1 < n/2$, we see, that it behaves like $-$ONEMAX with respect to STATIC EA. So, if p_j^+ resp. p_j^- is the probability of increasing the number of ones resp. decreasing the number of ones by one, when the actual x contains exactly j ones, we have for all $j \in \{0,\ldots,n/2-1\}$

$$p_j^+ = \frac{n-j}{\alpha \cdot n} \text{ and } p_j^- = \frac{j}{n}.$$

Hence, using Corollary 2.4, we get for $E(T_i^+)$, the expected number of steps until we reach a bit string with $i+1$ ones, when starting with a bit string with $i < n/2$ ones:

$$\begin{aligned} E(T_i^+) &= \sum_{k=0}^{i} \frac{1}{p_k^+} \cdot \prod_{l=k+1}^{i} \frac{p_l^-}{p_l^+} = \sum_{k=0}^{i} \frac{\alpha \cdot n}{n-k} \cdot \prod_{l=k+1}^{i} \frac{l}{n} \cdot \frac{\alpha \cdot n}{n-l} \\ &= \sum_{k=0}^{i} \alpha^{i-k+1} \cdot \frac{n}{n-k} \cdot \frac{i! \cdot (n-i-1)!}{k! \cdot (n-k-1)!} = \sum_{k=0}^{i} \alpha^{i-k+1} \cdot \frac{\binom{n}{k}}{\binom{n-1}{i}}. \end{aligned} \quad (1)$$

So $E(T^+_{n/2-1})$ can be lower bounded in the following way

$$E(T^+_{n/2-1}) \;=\; \sum_{k=0}^{n/2-1} \alpha^{n/2-k} \cdot \frac{\binom{n}{k}}{\binom{n-1}{n/2-1}} \geq \frac{\alpha^{n/2}}{\binom{n-1}{n/2-1}} \geq \frac{\alpha^{n/2}}{2^n}. \tag{2}$$

Hence, $E(T^+_{n/2-1})$ is $\Omega\left(\left(\sqrt{\alpha/4}\right)^n\right)$. So for $\alpha \geq 4+\varepsilon$ (where $\varepsilon > 0$) this results in an exponential lower bound for $E(T^+_{n/2-1})$, implying this bound for $E(T_i)$ for all $i \in \{0, \ldots, n/2-1\}$. Because this is a constant fraction of all bit strings, we have an exponential lower bound for the expected number of steps of the whole Static EA for $\alpha \geq 4+\varepsilon$.

In the following we want to show an exponential lower bound for the expected number of steps $E(T_{n-1})$ for $\alpha < 4+\varepsilon$. When α is small, worsenings are accepted with large probability, so that we can expect $E(T_{n-1})$ to be large. To lower bound $E(T_{n-1})$ we use Lemma 2.3. Because we have for all $i \in \{n/2+2, \ldots, n-1\}$:

$$p_i^+ = \frac{n-i}{n} \text{ and } p_i^- = \frac{i}{n \cdot \alpha},$$

we can lower bound $E(T^+_{n-1})$ by:

$$\begin{aligned} E(T^+_{n-1}) \;&\geq\; \sum_{k=0}^{n/2-3} \frac{\prod_{l=0}^{k-1} p^-_{n-1-l}}{\prod_{l=0}^{k} p^+_{n-1-l}} = \sum_{k=0}^{n/2-3} \frac{\prod_{l=n-k}^{n-1} p_l^-}{\prod_{l=n-k-1}^{n-1} p_l^+} = \sum_{k=0}^{n/2-3} \frac{\prod_{l=n-k}^{n-1} \frac{l}{n\cdot\alpha}}{\prod_{l=n-k-1}^{n-1} \frac{n-l}{n}} \\ &= \sum_{k=0}^{n/2-3} \frac{n}{\alpha^k} \cdot \frac{(n-1)!}{(n-k-1)! \cdot (k+1)!} = \sum_{k=0}^{n/2-3} \frac{\binom{n}{k+1}}{\alpha^k} = \alpha \cdot \sum_{k=1}^{n/2-2} \frac{\binom{n}{k}}{\alpha^k} \\ &\geq\; \alpha \cdot \left(\frac{\left(\frac{1}{\alpha}+1\right)^{n-4}}{2} - 1 \right) = \Omega\left(\left(\frac{1}{\alpha}+1\right)^{n-4} \right). \end{aligned}$$

Hence, for all $i \in \{0, \ldots, n/2-1\}$ the expected value of T_i is

$$E(T_i) = \Omega\left(\left(\sqrt{\frac{\alpha}{4}}\right)^n + \left(\frac{1}{\alpha}+1\right)^{n-4} \right),$$

which is exponential for all choices of $\alpha \in [1; \infty[$. As the fraction of bit strings with at most $n/2-1$ ones is constant, the expected running time of Static EA is exponential for all α. □

Intuitively, Dynamic EA can perform better than Static EA on Valley, if the selection schedule works as follows: in the beginning, worsenings are accepted with probability almost one, so that the actual point x is making a random walk, until its number of ones rises to $n/2+1$. After this point the probability of accepting worsenings should decrease, so that only improvements are accepted in order to reach the maximum $(1, \ldots, 1)$ quickly.

These intuitive concepts will be proven rigorously in the next theorem.

Theorem 4.3: *With probability* $1-\mathrm{O}\left(n^{-n}\right)$ *the number of steps until* Dynamic EA *with the selection schedule*

$$\alpha(t) := 1 + \frac{t}{s(n)}$$

reaches the maximum of Valley *for the first time is* $\mathrm{O}\left(n \cdot s(n)\right)$ *for any polynomial* s *with* $s(n) \geq 2en^4 \log n$. *Furthermore, the expected number of steps until this happens is* $\mathrm{O}\left(n \cdot s(n)\right)$, *if we additionally have* $\alpha(t) = 1$ *for* $t > 2^n$.

Proof: The basic idea of the proof is similar to the proof of Theorem 3.4. We split the run of Dynamic EA into two phases of predefined length. We show that with very high probability a state with at least $n/2+1$ ones is reached within the first phase, and all succeeding states have at least $n/2+1$ ones, too. Furthermore, with very high probability the optimum is reached within the second phase. Finally, we upper bound the expected number of steps in the case, that any of these events do not happen.

The first phase has length $s(n)/n + 2en^3 \log n$. We want to upper bound the expected number of steps in the first phase Dynamic EA takes to reach a state with at least $n/2+1$ ones. For that purpose we upper bound $\mathrm{E}\left(T_i^+\right)$ for all $i \in \{0, \ldots, n/2\}$. We do not care what happens during the first $s(n)/n$ steps. After that, we have $\alpha(t) \geq 1 + 1/n$. Pessimistically we assume that the current state at step $t = s(n)/n$ contains at most $n/2$ ones.

We use equation (1) of Theorem 4.2, which is valid for $i \in \{0, \ldots, n/2+1\}$.

$$\begin{aligned}
\mathrm{E}\left(T_i^+\right) &= \sum_{j=0}^{i} \alpha^{i-j+1} \cdot \frac{\binom{n}{j}}{\binom{n-1}{i}} = \sum_{j=0}^{i} \alpha^{j+1} \cdot \frac{\binom{n}{i-j}}{\binom{n-1}{i}} \\
&= \sum_{j=0}^{i} \alpha^{j+1} \cdot \frac{n!}{(i-j)! \cdot (n-i+j)!} \cdot \frac{i! \cdot (n-1-i)!}{(n-1)!} = \sum_{j=0}^{i} \alpha^{j+1} \cdot \frac{\binom{i}{j}}{\binom{n-i+j}{j}} \cdot \frac{n}{n-i}
\end{aligned}$$

As the last expression decreases with decreasing i, it follows that $\mathrm{E}\left(T_i^+\right) \leq \mathrm{E}\left(T_{i+1}^+\right)$ for all $i \in \{0, \ldots, n/2-1\}$. Since the length of the first phase is $s(n)/n + 2en^3 \log n$, we have $\alpha(t) \leq 1 + 2/n$ during the first phase. Using this and setting $i = n/2-1$, we get

$$\mathrm{E}\left(T_{n/2-1}^+\right) \leq \sum_{j=0}^{n/2-1} \left(1 + \frac{2}{n}\right)^{j+1} \frac{\binom{n/2-1}{j}}{\binom{n/2+1+j}{j}} \cdot \frac{n}{n/2+1} \leq 2 \sum_{j=0}^{n/2-1} e = en.$$

Hence, by using Lemma 2.2 we can upper bound $\mathrm{E}\left(T_{n/2}^+\right)$ by

$$\mathrm{E}\left(T_{n/2}^+\right) = \frac{1}{(n/2)/n} + \frac{(n/2)/n}{(n/2)/n} \cdot \mathrm{E}\left(T_{n/2-1}^+\right) \leq 2 + en.$$

So, the expected number of steps until a bit string with more than $n/2$ ones is reached is bounded above by

$$(n/2) \cdot en + 2 + en \leq en^2.$$

We use the Markov inequality and see, that the probability of not reaching a state with more than $n/2$ ones within $2en^2$ steps is at most $1/2$. Our analysis is independent of the current bit string at the beginning of such a subphase of length $2en^2$. So, we can consider the $2en^3 \log n$ steps in the first phase as $n \log n$ independent subphases of length $2en^2$ each. Hence, the probability of not reaching a state with more than $n/2$ ones within the first phase is $\mathrm{O}\left(n^{-n}\right)$.

Assume that DYNAMIC EA reaches a bit string with more than $n/2$ ones at some step t with $t \geq s(n)/n$. This yields $\alpha(t) \geq 1 + 1/n$. Let $p(n)$ be some polynomial. The probability to reach a bit string with at most $n/2$ ones within $p(n)$ steps is bounded above by

$$p(n) \cdot \frac{n/2+1}{n \cdot (1+1/n)^{7n^2 \ln n}} < \frac{p(n)}{e^{4n \ln n}} = \mathrm{O}\left(n^{-n}\right)$$

where the last equality follows since $p(n)$ is a polynomial. We conclude that after once reaching a bit string with more than $n/2$ ones, for polynomially bounded number of steps the number of ones is larger than $n/2$, too, with probability $1 - \mathrm{O}\left(n^{-n}\right)$. Hence, after the first phase the probability of not being in a state with more than $n/2$ ones is $\mathrm{O}\left(n^{-n}\right)$.

Now we consider the succeeding second phase, which ends with $t = n \cdot s(n)$, which is polynomially bounded. Therefore, we neglect the case that during the second phase a bit string with at most $n/2$ ones is reached. We saw above, that this case has probability $\mathrm{O}\left(n^{-n}\right)$.

We want to prove that with very high probability the optimum is reached within the second phase. In order to do so we upper bound the expected number of steps DYNAMIC EA needs to reach the optimum. We do not care about the beginning of phase 2 and consider only steps with $t \geq (n-1)s(n)$. Then we have $\alpha(t) \geq n$. Due to the length of the second phase, we have $\alpha(t) \leq n+1$, too. Using equation (2) of Theorem 4.2, we can upper bound $\mathrm{E}\left(T^+_{(n/2)-1}\right)$ in the following way.

$$\mathrm{E}\left(T^+_{n/2-1}\right) \leq \sum_{j=0}^{n/2-1} (n+1)^{n/2-j} \cdot \frac{\binom{n}{j}}{\binom{n-1}{n/2-1}} \leq \sum_{j=0}^{n/2-1} \binom{n}{j} n^{n-j} = \frac{(1+n)^n}{2}$$

Hence, we can upper bound $\mathrm{E}\left(T^+_{n/2}\right)$ by

$$\mathrm{E}\left(T^+_{n/2}\right) = \frac{1}{(n/2)/n} + \frac{(n/2)/n}{(n/2)/n} \cdot \mathrm{E}\left(T^+_{n/2-1}\right) \leq 2 + \frac{(1+n)^n}{2}$$

and $\mathrm{E}\left(T^+_{n/2+1}\right)$ for $n \geq 3$ by

$$\begin{aligned}
\mathrm{E}\left(T^+_{n/2+1}\right) &\leq \frac{1}{(n/2-1)/n} + \frac{(n/2+1)/(n \cdot n^{7n^2 \ln n})}{(n/2-1)/n} \cdot \mathrm{E}\left(T^+_{n/2}\right) \\
&\leq \frac{2n}{n-2} + \frac{n+2}{2n^{7n^2 \ln(n)+1}} \cdot \frac{2n}{n-2}(1+n)^n \\
&\leq 6 + \frac{5(1+n)^n}{n^{7n^2 \ln(n)+1}} \leq 7.
\end{aligned}$$

Using Lemma 2.3 for $j = i - n/2 - 1$, we get for all $i \in \{n/2+2, \dots, n-1\}$

$$\begin{aligned}
\mathrm{E}\left(T_i^+\right) &= \left(\sum_{j=0}^{i-n/2-2} \frac{\prod_{k=0}^{j-1} p_{i-k}^-}{\prod_{k=0}^{j} p_{i-k}^+}\right) + \frac{\prod_{k=0}^{i-n/2-2} p_{i-k}^-}{\prod_{k=0}^{i-n/2-2} p_{i-k}^+} \cdot \mathrm{E}\left(T_{n/2+1}^+\right) \\
&\leq \left(\sum_{j=0}^{i-n/2-2} \frac{\prod_{k=i-j+1}^{i} p_k^-}{\prod_{k=i-j}^{i} p_k^+}\right) + \frac{\prod_{k=n/2+2}^{i} p_k^-}{\prod_{k=n/2+2}^{i} p_k^+} \cdot 7.
\end{aligned}$$

As Valley behaves like OneMax for all states with at least $n/2+2$ ones with respect to Dynamic EA, we have $p_k^+ = (n-k)/n$ and $p_k^- \leq k/(n\alpha)$. Hence, we get

$$\begin{aligned}
\mathrm{E}\left(T_i^+\right) &\leq \left(\sum_{j=0}^{i-n/2-2} \frac{\prod_{k=i-j+1}^{i} k/(n \cdot n)}{\prod_{k=i-j}^{i} (n-k)/n}\right) + \frac{\prod_{k=n/2+2}^{i} k/(n \cdot n)}{\prod_{k=n/2+2}^{i} (n-k)/k} \cdot 7 \\
&= \left(\sum_{j=0}^{i-n/2-2} \frac{n^{-2j} \cdot i!/(i-j)!}{n^{-j-1} \cdot (n-i+j)!/(n-i-1)!}\right) + \\
&\qquad \frac{n^{2i+n+2} \cdot i!/(n/2+1)!}{n^{-i+n/2+1} \cdot (n/2-2)!/(n-i-1)!} \cdot 7 \\
&= \left(\sum_{j=0}^{i-n/2-2} n^{1-j} \cdot \frac{\binom{i}{j}}{(n-i) \cdot \binom{n-i+j}{j}}\right) + \frac{(n/2-1) \cdot \binom{n}{n/2+1} \cdot n^{n/2+1}}{(n-i) \cdot \binom{n}{i} \cdot n^i} \cdot 7.
\end{aligned}$$

To upper bound the second term, we derive the following for all $i \in \{0, \dots, n-2\}$.

$$\begin{aligned}
& (n-i) \cdot \binom{n}{i} \cdot n^i \leq (n-(i+1)) \cdot \binom{n}{i+1} \cdot n^{i+1} \\
\iff \quad & \frac{n-i}{n-i-1} \cdot \frac{n!}{i! \cdot (n-i)!} \cdot \frac{(i+1)! \cdot (n-i-1)!}{n!} \leq n \\
\iff \quad & \frac{i+1}{n-i-1} \leq n, \text{ which is valid for all } i \in \{0, \dots, n-2\}.
\end{aligned}$$

Hence, we get the following upper bound for $\mathrm{E}\left(T_i^+\right)$, as i is at least $n/2+2$:

$$\mathrm{E}\left(T_i^+\right) \leq \left(\sum_{j=0}^{i-n/2-2} n^{1-j} \cdot \frac{\binom{i}{j}}{(n-i) \cdot \binom{n-i+j}{j}}\right) + 7.$$

So, by upper bounding $\mathrm{E}\left(T_{n-1}^+\right)$, we get an upper bound for $\mathrm{E}\left(T_i^+\right)$ for all $i \in \{n/2+2, \dots, n-1\}$ (and $n \geq 3$).

$$\begin{aligned}
\mathrm{E}\left(T_{n-1}^+\right) &\leq n \cdot \left(\sum_{j=0}^{n/2-3} \left(\frac{1}{n}\right)^j \cdot \frac{\binom{n-1}{j}}{\binom{j+1}{j}}\right) + 7 \leq n \cdot \left(\sum_{j=0}^{n/2-3} \left(\frac{1}{n}\right)^j \binom{n}{j}\right) + 7 \\
&\leq n \cdot \frac{(1+1/n)^n}{2} + 7 \leq \frac{en}{2} + 7 \leq 4n
\end{aligned}$$

Hence, for all $i \in \{n/2+1, \dots, n\}$ the value of $\mathrm{E}(T_i)$ can be upper bounded by $2n^2$. Using the Markov inequality, this implies that after $4n^2$ steps the probability that the optimum is not reached is upper bounded by $1/2$. Considering the $s(n) \geq 2en^4 \log n$ steps as at least $en^2 \log n$ independent subphases of length $2n^2$ each, implies that the optimum is reached with probability $1 - \mathrm{O}(n^{-n})$. Altogether we proved that the optimum is reached within the first $n \cdot s(n)$ steps with probability $1 - \mathrm{O}(n^{-n})$.

In order to derive the upper bound on the expected number of steps we consider the case that the optimum is not reached. This has probability $\mathrm{O}(n^{-n})$. We use the additional assumption that $\alpha(t) = 1$ holds for $t > 2^n$. We do not care what else happens until $t > 2^n$ holds. Then we have $\alpha(t) = 1$. This implies that DYNAMIC EA performs a pure random walk. As we saw in the proof of Theorem 3.4, the expected number of steps in this case is upper bounded by $3 \cdot 2^n$. So, this case adds only

$$3 \cdot 2^n \cdot \mathrm{O}\left(n^{-n}\right) = \mathrm{O}(1)$$

to the expected running time. Altogether, we see that the expected running time is upper bounded by $\mathrm{O}(n \cdot s(n))$. □

5 Conclusions

We presented a simple evolutionary algorithm, which accepts worsenings with some probability. In order to see how even simple dynamic parameter control mechanisms can speed up the search process significantly, we compared it to a variant where the selection probability varies over the number of steps made so far. We analyzed the expected running times of these two EAs for two different objective functions, and exhibited that they are exponential for any constant selection probability but only polynomial for appropriate time-dependent selection probabilities. Furthermore, we have shown that for these time-dependent selection probabilities the probability of using more than polynomially many steps is exponentially small. Hence, we have given a proof that dynamic parameter control in EAs can help significantly. It is an open problem to give similar examples for the other classes of non-static parameter settings, namely adaptive and self-adaptive parameter control.

We note, that the algorithms are instances of simulated annealing respectively the Metropolis algorithm. So our approach is also a step towards answering the still open question, whether simulated annealing with a natural cooling schedule outperforms the Metropolis algorithm with a carefully chosen temperature for some natural problem. We believe that presenting carefully designed and well structured, although artificial functions with desired properties, helps to understand the way the two algorithms achieve optimization. Furthermore, we believe that this understanding is helpful if not even necessary to identify a natural problem where the two algorithms show different behavior.

References

T. Bäck (1998). An overview of parameter control methods by self-adaptation in evolutionary algorithms. *Fundamenta Informaticae 34*, 1–15.

S. Droste, T. Jansen, and I. Wegener (1998a). On the analysis of the $(1+1)$ evolutionary algorithm. Technical Report CI-21/98, University of Dortmund, Reihe Computational Intelligence, Collaborative Research Center 531. Submitted to: Theoretical Computer Science.

S. Droste, T. Jansen, and I. Wegener (1998b). On the optimization of unimodal functions with the (1+1) evolutionary algorithm. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel (Eds.), *Parallel Problem Solving from Nature (PPSN V)*, Berlin, Germany, 47–56. Springer.

S. Droste, T. Jansen, and I. Wegener (1998c). A rigorous complexity analysis of the $(1+1)$ evolutionary algorithm for separable functions with Boolean inputs. *Evolutionary Computation 6*(2), 185–196.

D. B. Fogel (1995). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press.

T. Hagerup and C. Rüb (1989). A guided tour of Chernoff bounds. *Information Processing Letters 33*, 305–308.

J. H. Holland (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan.

M. Jerrum and A. Sinclair (1997). The Markov chain Monte Carlo method: an approach to approximate counting and integration. In D. S. Hochbaum (Ed.), *Approximation Algorithms for NP-hard Problems*, 482–520. Boston, MA: PWS Publishers.

M. Jerrum and G. B. Sorkin (1998). The Metropolis algorithm for graph bisection. *Discrete Applied Mathematics 82*, 155–175.

J. R. Koza (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculation by fast computing machines. *Journal of Chemical Physics 21*, 1087–1092.

G. Rudolph (1997). *Convergence Properties of Evolutionary Algorithms*. Hamburg, Germany: Verlag Dr. Kovač.

H.-P. Schwefel (1995). *Evolution and Optimum Seeking*. New-York, NY: Wiley.

A. Sinclair (1993). *Algorithms for Random Generation & Counting*. Bosten: Birkhäuser.

G. B. Sorkin (1991). Efficient simulated annealing on fractal energy landscapes. *Algorithmica 6*, 367–418.