Analysis of Diversity-Preserving Mechanisms for Global Exploration*

Tobias Friedrich tobias.friedrich@mpi-inf.mpg.de
Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany

Pietro S. Oliveto P.S.Oliveto@cs.bham.ac.uk
University of Birmingham, Birmingham B15 2TT, United Kingdom

Dirk Sudholt dirk.sudholt@cs.tu-dortmund.de
Technische Universität Dortmund, 44221 Dortmund, Germany and
International Computer Science Institute, Berkeley, CA 94704, USA

Carsten Witt carsten.witt@cs.tu-dortmund.de
Technische Universität Dortmund, 44221 Dortmund, Germany and
Technical University of Denmark, 2800 Kgs. Lyngby, Denmark

Abstract

Maintaining diversity is important for the performance of evolutionary algorithms. Diversity-preserving mechanisms can enhance global exploration of the search space and enable crossover to find dissimilar individuals for recombination. We focus on the global exploration capabilities of mutation-based algorithms. Using a simple bimodal test function and rigorous runtime analyses, we compare well-known diversity-preserving mechanisms like deterministic crowding, fitness sharing, and others with a plain algorithm without diversification. We show that diversification is necessary for global exploration, but not all mechanisms succeed in finding both optima efficiently. Our theoretical results are accompanied by additional experiments for different population sizes.

Keywords

Diversity, runtime analysis, fitness sharing, deterministic crowding, exploration.

1 Introduction

In evolutionary computation the term *diversity* indicates dissimilarities of individuals and is considered an important property. In a population-based evolutionary algorithm without a diversity-preserving mechanism there is a risk of the best individual taking over the whole population before the fitness landscape is explored properly. When the population becomes completely redundant, the algorithm basically reduces to a trajectory-based algorithm while still suffering from high computational effort and space requirements for the whole population.

Diversity-preserving mechanisms can help the optimization in two ways. On one hand, a diverse population is able to deal with multimodal functions and can explore

* A preliminary version of this article appeared in [4].

Tobias Friedrich and Dirk Sudholt were partially supported by postdoctoral fellowships from the German Academic Exchange Service. Pietro S. Oliveto was supported by an EPSRC grant (EP/C520696/1). Dirk Sudholt and Carsten Witt were partially supported by the Deutsche Forschungsgemeinschaft (DFG) as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

several hills in the fitness landscape simultaneously. Diversity-preserving mechanisms can therefore support global exploration and help to locate several local and global optima. This behavior is welcome in dynamic environments as the algorithm is more robust with respect to changes of the fitness landscape. Moreover, the algorithm can offer several good solutions to the user, a feature desirable in multiobjective optimization. On the other hand, a diverse population gives higher chances to find dissimilar individuals and to create good offspring by recombining different "building blocks". Diversity-preserving mechanisms can thus enhance the performance of crossover.

Up to now, the use of diversity-preserving mechanisms has been assessed mostly by means of empirical investigations [e. g., 1, 20]. Theoretical runtime analyses have mostly used diversity-preserving mechanisms to enhance the performance of crossover. Jansen and Wegener [10] presented the first proof that crossover can make a difference between polynomial and exponential expected optimization times. They used a very simple diversity-preserving mechanism that only shows up as a tie-breaking rule: when there are several individuals with worst fitness among parents and offspring, the algorithm removes those individuals with a maximal number of genotype duplicates. Nevertheless, this mechanism makes the individuals spread on a certain fitness level (i. e., makes them produce different individuals of the same fitness) such that crossover is able to find suitable parents for recombination. Storch and Wegener [17] presented a similar result for populations of constant size. They used a stronger mechanism that prevents duplicates from entering the population, regardless of their fitness value.

Niching methods encourage the exploration of "niches", that is, they aim at the survival of individuals far apart from the other individuals. The first theoretical runtime analysis considering niching methods was presented by Fischer and Wegener [2] for a fitness function derived from a generalized Ising model on ring graphs. The authors compare the well-known (1+1) EA with a (2+2) GA with fitness sharing. Fitness sharing [11] derates the real fitness of an individual x by a measure related to the similarity of x to all individuals in the population, hence encouraging the algorithm to decrease similarity in the population. Fischer and Wegener prove that their genetic algorithm outperforms the (1+1) EA by a polynomial factor. Sudholt [18] extended this study for the Ising model on binary trees, where the performance gap between GAs and EAs is even larger. While a broad class of (μ+λ) EAs has exponential expected optimization time, a (2+2) GA with fitness sharing finds a global optimum in expected polynomial time.

In all these studies diversity is used to assist crossover. Contrarily, Friedrich, Hebbinghaus, and Neumann [3] focused on diversity-preserving mechanisms as a means to enhance the global exploration of EAs without crossover. Using rigorous runtime analyses, the authors compare a mechanism avoiding genotype duplicates with a strategy avoiding duplicate fitness values to spread individuals on different fitness levels. It is shown for artificial functions that both mechanisms can outperform one another drastically.

Friedrich et al. [3] were the first to focus on the use of diversity-preserving mechanisms for global exploration with respect to rigorous runtime analyses. However, their test functions are clearly tailored towards one particular diversity-preserving mechanism. We want to obtain a broader perspective including a broader range of diversity-preserving mechanisms. Therefore, we compare several well-known diversity-preserving mechanisms on the simplest bimodal function that may also appear as part of a real-world problem. On the considered function, simple hill climbers

find the global optimum with constant probability, hence a restart strategy is sufficient for optimization. We focus on well-known diversity-preserving mechanisms that do not restart the algorithm. Firstly, we rigorously prove that diversity-preserving mechanisms are necessary for our function since populations of almost linear size without diversification fail to find both peaks, with high probability. Then we analyze common diversity-preserving mechanisms and show that not all of them are effective for avoiding premature convergence even for such a simple landscape. As a result, we hope to get a more objective and more general impression of the capabilities and limitations of common diversity-preserving mechanisms.

This paper extends its conference version [4] in several ways. On one hand, some theoretical results have been improved. In particular, the negative result for fitness duplicates (Theorem 3) now holds for larger population sizes μ and the lower bound on the runtime has been improved by using stronger drift results. On the other hand, we supplement our theoretical work with experimental results. This enables us to compare our asymptotic theoretical predictions against empirical data for a concrete problem dimension and several values for the population size μ. Moreover, we consider the effect of very large μ to see whether for the diversity-preserving mechanisms very large populations can be successful where small populations fail.

In the remainder of this paper, we first present our bimodal test function in Section 2. Negative results for a plain (μ+1) EA in Section 3 show that diversification is needed. In Sections 4 and 5 we investigate the strategies previously analyzed by Friedrich et al. [3] to avoid genotype duplicates and fitness duplicates, respectively. Section 6 deals with the well-known deterministic crowding strategy [11] where offspring directly compete with their associated parents. Fitness sharing, which turns out to be the strongest mechanism, is analyzed in Section 7. Section 8 contains experimental results showing how well our theoretical results match with empirical results and revealing additional insight on the dynamic behavior of the algorithms. We present our conclusions in Section 9.

2 A Simple Bimodal Function

We consider a simple bimodal function called TWOMAX that has already been investigated in the context of genetic algorithms by Pelikan and Goldberg [16] and Goldberg, Van Hoyweghen, and Naudts [5]. The function TWOMAX is essentially the maximum of ONEMAX and ZEROMAX. Local optima are solutions 0^n and 1^n where the number of zeros or the number of ones, respectively, is maximized. Hence, TWOMAX can be seen as a bimodal equivalent of ONEMAX. The fitness landscape consists of two hills with symmetric slopes. In contrast to Pelikan and Goldberg [16] and Goldberg et al. [5] we modify the function slightly such that only one hill contains the global optimum, while the other one leads to a local optimum. This is done by simply adding an additional fitness value for 1^n, turning it into a unique global optimum. Hence, an unbiased random search heuristic cannot tell in advance which hill is more promising.

For $x = x_1 x_2 \dots x_n$, let $|x|_1 := \sum_{i=1}^{n} x_i$ denote the number of 1-bits and $|x|_0 := \sum_{i=1}^{n} (1 - x_i)$ denote the number of 0-bits in x. Then

$$\text{TWOMAX}(x) := \max\{|x|_0, |x|_1\} + \prod_{i=1}^{n} x_i.$$

Figure 1 shows a sketch of TWOMAX. Among all search points with more than $n/2$ 1-bits, the fitness increases with the number of ones. Among all search points with less

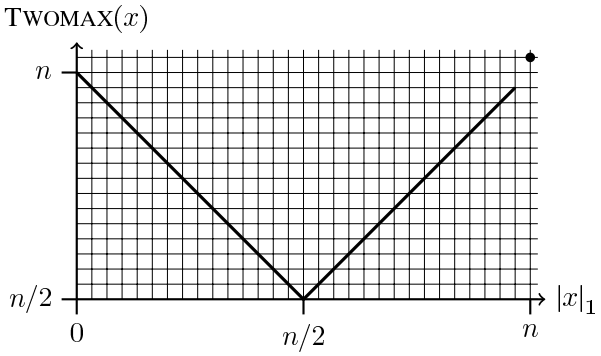

Figure 1: Sketch of TWOMAX. The dot indicates the global optimum.

than $n/2$ 1-bits, the fitness increases with the number of zeros. We refer to these sets as *branches* and the algorithms as climbing these two branches of TWOMAX.

The TWOMAX function does appear in well-known combinatorial optimization problems. For example, the VERTEXCOVER bipartite graph analyzed in Oliveto, He, and Yao [15] consists of two branches, one leading to a local optimum and the other to the minimum cover. In fact similar proof techniques as those used in this paper have also been applied in the VERTEXCOVER analysis of the (μ+1) EA for the bipartite graph. Another function with a similar structure is the MINCUT instance from Sudholt [19].

Considering μ independent parallel runs of the (1+1) EA (or μ starts of the algorithm with a suitable restart scheme) yields that the probability of not finding the global optimum efficiently is $2^{-\mu}$. When using populations of size μ, we expect the search to be more focused than with μ independent runs. Hence without diversification the probability of finding the optimum should be significantly worse. However, from a good diversity-preserving mechanism we expect that the probability of not finding the optimum decreases significantly with growing μ. We will see in the following that diversity-preserving mechanisms avoiding duplicates do not fulfil this property and that the probability of finding the global optimum efficiently is at most $1/2 + o(1)$ (see Motwani and Raghavan [12] for the asymptotic notation), hence comparable to the simple (1+1) EA. We can conclude for these mechanisms that populations are nearly useless on TWOMAX. On the other hand, deterministic crowding shows a behavior very similar to μ independent runs and fitness sharing even finds the optimum efficiently with probability 1.

Note that optimizing TWOMAX is not always inefficient. All algorithms considered hereinafter are able to find either 0^n or 1^n efficiently, hence there is still a good chance that the global optimum is found in short time. For the plain (μ+1) EA the expected time to find 0^n or 1^n is bounded by the expected optimization time on ONEMAX, hence at most $O(\mu n + n \log n)$ [22]. For all variants of the (μ+1) EA the weaker bound $\mu en(\ln n + 1) = O(\mu n \log n)$ holds. This bound relies on the fact that 1-bit mutations can effectively decrease the distance to the closest point from $\{0^n, 1^n\}$ if an elitist is chosen as parent. Due to symmetry, the global optimum 1^n is found first with probability $1/2$. This implies that appropriate restart strategies yield an expected optimization time of $O(\mu n \log n)$. In the following, we will, however, consider algorithms without restarts and prove for some diversity-preserving mechanisms that the global optimum is not found in exponential time with probability close to $1/2$.

3 No Diversity-Preserving Mechanism

In order to obtain a fair comparison of different diversity-preserving mechanisms, we keep one algorithm fixed as much as possible. The basic algorithm, the following (μ+1) EA, has already been investigated by Witt [22].

Algorithm 1 (μ+1) EA

Let $t := 0$ and initialize P_0 with μ individuals chosen uniformly at random.
repeat
  Choose $x \in P_t$ uniformly at random.
  Create y by flipping each bit in x independently with probability $1/n$.
  Choose $z \in P_t$ with worst fitness uniformly at random.
  if $f(y) \geq f(z)$ **then** $P_{t+1} = P_t \setminus \{z\} \cup \{y\}$ **else** $P_{t+1} = P_t$.
  Let $t = t + 1$.

The (μ+1) EA uses random parent selection and elitist selection for survival. As parents are chosen randomly, the selection pressure is quite low. Nevertheless, the (μ+1) EA is not able to maintain individuals on both branches for a long time. We now show that if μ is not too large, the individuals on one branch typically get extinct before the top of the branch is reached. Thus, the global optimum is found only with probability close to $1/2$ and the expected optimization time is very large.

Theorem 1. *The probability that the (μ+1) EA with no diversity-preserving mechanism and $\mu = o(n/\log n)$ optimizes* TWOMAX *in time n^{n-1} is at most $1/2 + o(1)$. Its expected optimization time is $\Omega(n^n)$.*

Proof. The probability that during initialization either 0^n or 1^n is created is bounded by $\mu \cdot 2^{-n+1}$, hence exponentially small. In the following, we assume that such an atypical initialization does not happen as this assumption only introduces an error probability of $o(1)$.

Consider the algorithm at the first point of time T^* where either 0^n or 1^n is created. Due to symmetry, the local optimum 0^n is created with probability $1/2$. We assume in the following that 0^n is created and keep in mind an error probability of $1/2$. We now show that then with high probability 0^n takes over the population before the global optimum 1^n is created. Let i be the number of copies of 0^n in the population. From the perspective of extinction, a good event G_i is to increase this number from i to $i+1$. For $n \geq 2$ we have $\mathrm{P}(G_i) \geq \frac{i}{\mu} \cdot \left(1 - \frac{1}{n}\right)^n \geq \frac{i}{4\mu}$ since it suffices to select one out of i copies and to create another copy of 0^n. On the other hand, the bad event B_i is to create 1^n in one generation. This probability is maximized if all $\mu - i$ remaining individuals contain $n-1$ ones: $\mathrm{P}(B_i) \leq \frac{\mu - i}{\mu} \cdot \frac{1}{n} < \frac{1}{n}$. Together, the probability that the good event G_i happens before the bad event B_i is

$$\mathrm{P}(G_i \mid G_i \cup B_i) \;\geq\; \frac{\mathrm{P}(G_i)}{\mathrm{P}(G_i) + \mathrm{P}(B_i)} \;\geq\; \frac{i/(4\mu)}{i/(4\mu) + 1/n} \;=\; 1 - \frac{1/n}{i/(4\mu) + 1/n} \;\geq\; 1 - \frac{4\mu}{in}.$$

The probability that 0^n takes over the population before the global optimum is reached is therefore bounded by

$$\prod_{i=1}^{\mu} \mathrm{P}(G_i \mid G_i \cup B_i) \;\geq\; \prod_{i=1}^{\mu} \left(1 - \frac{4\mu}{in}\right).$$

Using $4\mu/n \leq 1/2$ and $1 - x \geq e^{-2x}$ for $x \leq 1/2$, we obtain

$$\prod_{i=1}^{\mu}\left(1-\frac{4\mu}{in}\right) \geq \prod_{i=1}^{\mu}\exp\left(-\frac{8\mu}{in}\right) = \exp\left(-\frac{8\mu}{n}\cdot\sum_{i=1}^{\mu}\frac{1}{i}\right)$$
$$\geq \exp(-O((\mu\log\mu)/n)) \geq 1 - O((\mu\log\mu)/n) = 1 - o(1).$$

If the population only contains copies of 0^n, mutation has to flip all n bits to reach the global optimum. This event has probability n^{-n} and, by the union bound, the probability of this happening in a phase consisting of n^{n-1} generations is at most $1/n$. The sum of all error probabilities is $1/2 + o(1)$, which proves the first claim.

For the second claim, observe that the conditional expected optimization time is n^n once the population has collapsed to copies of 0^n. As this situation occurs with probability at least $1/2 - o(1)$, the unconditional expected optimization time is $\Omega(n^n)$. □

4 No Genotype Duplicates

It has become clear that diversity-preserving mechanisms are very useful to optimize even a simple function such as TWOMAX. The simplest way to enforce diversity within the population is not to allow genotype duplicates. The following algorithm has been defined and analyzed by Storch and Wegener [17]. It prevents identical copies from entering the population as a natural way of ensuring diversity. We will, however, show that this mechanism is not powerful enough to explore both branches of TWOMAX.

Algorithm 2 (μ+1) EA with genotype diversity

Let $t := 0$ and initialize P_0 with μ individuals chosen uniformly at random.
repeat
  Choose $x \in P_t$ uniformly at random.
  Create y by flipping each bit in x independently with probability $1/n$.
  if $y \notin P_t$ **then**
    Choose $z \in P_t$ with worst fitness uniformly at random.
    if $f(y) \geq f(z)$ **then** $P_{t+1} = P_t \setminus \{z\} \cup \{y\}$ **else** $P_{t+1} = P_t$.
  Let $t = t + 1$.

We prove that if the population is not too large, the algorithm can be easily trapped in a local optimum.

Theorem 2. *The probability that the (μ+1) EA with genotype diversity and $\mu = o(n^{1/2})$ optimizes* TWOMAX *in time n^{n-2} is at most $1/2 + o(1)$. Its expected optimization time is $\Omega(n^{n-1})$.*

Proof. We use a similar way of reasoning as in the proof of Theorem 1. With probability $1/2 - o(1)$, 0^n is the first local or global optimum created at time T^*. Call x *good* (from the perspective of extinction) if $|x|_1 \leq 1$ and *bad* if $|x|_1 \geq n - 1$. At time T^* the number of good individuals is at least 1. In the worst case (again from the perspective of extinction) the population at time T^* consists of 0^n and $\mu - 1$ bad individuals with $n - 1$ ones. Provided that the (μ+1) EA does not flip $n - 2$ bits at once, we now argue that the number of good individuals is monotone unless the unique 0-bit in a bad individual is flipped.

Due to the assumptions on the population only offspring with fitness at least $n - 1$ are accepted, i. e., only good or bad offspring. In order to create a bad offspring, the

unique 0-bit has to be flipped since otherwise a clone or an individual with worse fitness is obtained. Hence the number of good individuals can only decrease if a bad individual is chosen as parent and its unique 0-bit is flipped. If there are i good individuals, we denote this event by B_i and have $\mathrm{P}(B_i) \leq \frac{\mu-i}{\mu} \cdot \frac{1}{n}$.

On the other hand, the number of good individuals is increased from i to $i+1$ if a good offspring is created and a bad individual is removed from the population. We denote this event by G_i. A good offspring is created with probability at least $1/(3\mu)$ for the following reasons. The point 0^n is selected with probability at least $1/\mu$ and then there are $n-(i-1) = n - o(n^{1/2}) \geq (e/3) \cdot n$ 1-bit mutations (provided n is large enough) creating good offspring that are not yet contained in the population. Along with the fact that a specific 1-bit mutation has probability $1/n \cdot (1-1/n)^{n-1} \geq 1/(en)$, the bound $1/(3\mu)$ follows. After creating such a good offspring, the algorithm removes an individual with fitness $n-1$ uniformly at random. As there are $i-1$ good individuals with this fitness and $\mu-i$ bad individuals, the probability of removing a bad individual equals $(\mu-i)/(\mu-1) \geq (\mu-i)/\mu$. Together, $\mathrm{P}(G_i) \geq \frac{1}{3\mu} \cdot \frac{\mu-i}{\mu} = \frac{\mu-i}{3\mu^2}$. The probability that G_i happens before B_i is at least

$$\mathrm{P}(G_i \mid G_i \cup B_i) \;\geq\; \frac{\frac{\mu-i}{3\mu^2}}{\frac{\mu-i}{3\mu^2} + \frac{\mu-i}{\mu n}} \;=\; \frac{1}{1+3\mu/n} \;=\; 1 - \frac{3\mu/n}{1+3\mu/n} \;\geq\; 1 - \frac{3\mu}{n}.$$

The probability that the number of good individuals increases to μ before the global optimum is reached is

$$\prod_{i=1}^{\mu} \mathrm{P}(G_i \mid G_i \cup B_i) \;\geq\; \left(1 - \frac{3\mu}{n}\right)^{\mu} \;\geq\; 1 - \frac{3\mu^2}{n} \;=\; 1 - o(1).$$

The probability of creating a global optimum, provided the population contains only search points with at most one 1-bit, is at most $n^{-(n-1)}$. The probability of this happening in a phase of n^{n-2} generations is still at most $1/n$. Adding up all error probabilities, the first claim follows.

The claim on the expected optimization time follows as the last situation is reached with probability at least $1/2 - o(1)$ and the conditional expected optimization time is at least n^{n-1} then. □

5 No Fitness Duplicates

Avoiding genotype duplicates does not help much to optimize TWOMAX as individuals from one branch are still allowed to spread on a certain fitness level and take over the population. A more restrictive mechanism is to avoid fitness duplicates, i. e., multiple individuals with the same fitness. Such a mechanism has been defined and analyzed by Friedrich et al. [3] for plateaus of constant fitness. In addition, this resembles the idea of fitness diversity proposed by Hutter and Legg [8].

The following (μ+1) EA with fitness diversity avoids that multiple individuals with the same fitness are stored in the population. If at some time t a new individual x is created with the same fitness value as a pre-existing one $y \in P_t$ then x replaces y.

From the analysis of Friedrich et al. [3] it can be derived that if the population size μ is a constant then the runtime on a simple plateau is exponential in the problem size n. Only if μ is very close to n the expected runtime is polynomial. In particular, if $\mu = n$ then the same upper bound as that of the (1+1) EA for plateaus of constant fitness [9] can be obtained (i. e., $O(n^3)$). In the following, by analyzing the mechanism

Algorithm 3 (μ+1) EA with fitness diversity

Let $t := 0$ and initialize P_0 with μ individuals chosen uniformly at random.
repeat
  Choose $x \in P_t$ uniformly at random.
  Create y by flipping each bit in x independently with probability $1/n$.
  if there exists $z \in P_t$ such that $f(y) = f(z)$
  then $P_{t+1} = P_t \setminus \{z\} \cup \{y\}$,
  else Choose $z \in P_t$ with worst fitness uniformly at random.
    if $f(y) \geq f(z)$ **then** $P_{t+1} = P_t \setminus \{z\} \cup \{y\}$ **else** $P_{t+1} = P_t$.
  Let $t = t + 1$.

on TWOMAX, we show how also on a simple bimodal landscape, fitness diversity does not help the (μ+1) EA to avoid getting trapped on a local optimum.

The following theorem proves that if the population is not too large, then with high probability the individuals climbing one of the two branches will be extinguished before any of them reaches the top. Since the two branches of the TWOMAX function are symmetric, this also implies that the global optimum will not be found in polynomial time with probability $1/2 - o(1)$.

Theorem 3. *The probability that the (μ+1) EA with fitness diversity and $\mu = \text{poly}(n)$ optimizes* TWOMAX *in time 2^{cn}, $c > 0$ being an appropriate constant, is at most $1/2 + o(1)$. Its expected optimization time is $2^{\Omega(n)}$.*

The upcoming proof of Theorem 3 investigates a potential function of the current population. The potential performs a random walk on a finite set of integers. In order to analyze first hitting times of this random walk, we need two technical results concerned with the so-called *drift*, i. e., the expected movement of the random walk in one step (see Oliveto, He, and Yao [14] for a general description of drift analysis applied to evolutionary algorithms).

If a random walk has a drift towards a target value, the expected first hitting time of the target value can be bounded from above using a bound on the drift. The first inequality of the following lemma has been proved by He and Yao [7], the second inequality follows from the first one using the law of total expectation. A similar result has been proved independently by Wegener and Witt [21].

Lemma 1 (Upper Bound, Drift towards Target)**.** *Consider a Markov process $\{X_t\}_{t\geq 0}$ with state space $\mathbb{N}_0$. Let $T := \inf\{t \geq 0 \colon X_t = 0\}$. If there exists $\delta > 0$ such that for any time $t \geq 0$ and any state $X_t > 0$ the condition $E(X_t - X_{t+1} \mid X_t) \geq \delta$ holds, then*

$$E(T \mid X_0) \leq \frac{X_0}{\delta} \qquad \text{and} \qquad E(T) \leq \frac{E(X_0)}{\delta}.$$

On the other hand, a random walk may have a drift leading away from the target value on an interval of the state space. If, additionally, large jumps are unlikely, the expected time to cross the interval is then bounded below by a value exponential in the length of the interval. Such a technical result has first been presented by Hajek [6] for the analysis of randomized search heuristics. The following major simplification has been obtained recently by Oliveto and Witt [13].

Lemma 2 (Lower Bound, Drift away from Target)**.** *Consider a Markov process $\{X_t\}_{t\geq 0}$ with state space $S = \{0, 1, \ldots, N\}$ for some $N \in \mathbb{N}$. Let $0 \leq a < b \leq N$ and define the*

stopping time $T = \min\{t \geq 0 \colon X_t \leq a \mid X_0 \geq b\}$. *For* $i \in S$ *and* $t \geq 0$ *let* $\Delta_t(i) := (X_{t+1} - X_t \mid X_t = i)$. *Suppose there are constants* $\delta, \varepsilon, r > 0$ *such that for all* $t \geq 0$

- $E(\Delta_t(i)) \geq \varepsilon$ *for* $a < i < b$ *and*
- $\mathrm{P}(\Delta_t(i) = -j) \leq 1/(1+\delta)^{j-r}$ *for* $i > a$ *and* $j \geq 1$,

then there is a constant $c > 0$ *such that* $\mathrm{P}\big(T \leq 2^{c(b-a)}\big) = 2^{-\Omega(b-a)}$.

Now we are prepared to prove Theorem 3.

Proof of Theorem 3. Obviously the second claim follows from the first one. W. l. o. g. $n/6 \in \mathbb{N}$. By Chernoff bounds, the probability that initialization creates a search point with at most $n/3$ 1-bits or at most $n/3$ 0-bits is at most $\mu \cdot 2^{-\Omega(n)} = o(1)$. We assume in the following that only search points x with $n/3 < |x|_1 < 2n/3$ are created and keep in mind an error probability of $o(1)$.

Let individuals with $i < n/2$ 0-bits be called x_i and individuals with $i < n/2$ 1-bits be called y_i. Initially there is neither x_i nor y_i in the population for $0 \leq i \leq n/3$. Because of the fitness diversity mechanism, there may be only one such x_i or one y_i in the population at the same time for $1 \leq i \leq n/3$ as x_i and y_i have the same fitness. For the current population P_t at time t define a potential $\varphi = \varphi(P_t)$ as follows:

$$\varphi \;=\; \varphi(P_t) \;:=\; \min\Big\{\min\{i \colon y_{i+1} \notin P_t\},\; \frac{n}{6}\Big\}.$$

The potential is capped at $n/6$ for technical reasons that will become obvious later on. As $n/6 < n/3$, we can conclude from potential i that $x_0, \ldots, x_i \notin P_t$. Intuitively, the potential is then a lower bound for the Hamming distance from the closest point in the population to the global optimum. We first consider the case $\mu \leq n/3$ as then we do not have to deal with the fact that the μ best fitness levels may contain multiple individuals. Later on, we then argue how to deal with larger population sizes.

We now partition the run into phases and call such a phase good (for the lower bound) if the global optimum is not found during the phase. Phase 1 ends when a search point in $\{0^n, 1^n\}$ is found for the first time. After the initialization only one new individual is created in each generation. Due to the symmetry of the TWOMAX function, the probability that 0^n is found first equals the probability that 1^n is found first. Hence Phase 1 is good with probability $1/2 - o(1)$.

At the end of Phase 1 we have a current potential of $\varphi \geq 1$. Let $m_1 := \min\{\mu - 1, \sqrt[3]{n}\}$ and $m_2 := \min\{\mu - 1, n/6\}$ be the maximum potential. Define Phase 2 to start after a good Phase 1 and to end when either the global optimum is found or the potential increases to $\varphi \geq m_1$.

We inspect changes of the potential. It may happen that the next step creates $y_{\varphi+1}$. Then $x_{\varphi+1}$ is removed from the population and φ increases by 1. Such a step is called a *good step*. On the other hand, if x_i for $i \leq \varphi$ is created then y_i is removed and φ decreases to $i - 1$. This is referred to as a *bad step*. All other steps do not change the potential. For proving a lower bound, given a current potential φ, we consider the following population as a worst case:

$$P_t \;=\; \{y_0, \ldots, y_\varphi,\; x_{\varphi+1}, \ldots, x_{\mu-1}\}.$$

In P_t we have $y_0, \ldots, y_\varphi \in P_t$ by definition of the potential φ and the x-individuals are stacked one after another. The latter maximizes the probability of reaching the global optimum in a single step. More general, for every $i \leq \varphi$ the probability of

creating x_i is maximized—this event is equivalent to a bad step decreasing the potential to i. Furthermore, the probability of a good step is minimized as all x-individuals have the largest possible Hamming distance to search points $y_{\varphi+1}$. Recall that every fitness level contains at most one search point, hence there cannot be a worse constellation for the x-individuals.

We now simplify the analysis by making pessimistic assumptions that reduce the algorithm towards a simple Markov chain. After a step has been made, we compute the new potential and then pessimistically replace the resulting population by the worst-case population corresponding to the new potential. Even in this pessimistic setting φ increases to m_1 with high probability before a bad step happens. If $x_{\varphi+j}$ is chosen as parent, a necessary condition for a bad step is that j out of $\varphi+j$ 0-bits flip. Using $\binom{n}{k} \leq (ne/k)^k$, the probability of a bad step is at most

$$\begin{aligned}
\frac{1}{\mu}\sum_{j=1}^{n/3-\varphi}\binom{j+\varphi}{j}\left(\frac{1}{n}\right)^j &\leq \frac{1}{\mu}\sum_{j=1}^{n/3-\varphi}\binom{j+m_1-1}{j}\left(\frac{1}{n}\right)^j \\
&\leq \frac{1}{\mu}\sum_{j=1}^{n/3-\varphi}\left(\frac{e(j+m_1-1)}{jn}\right)^j \\
&\leq \frac{1}{\mu}\sum_{j=1}^{\infty}\left(\frac{em_1}{n}\right)^j = \frac{1}{\mu}\cdot\frac{em_1}{n-em_1} < \frac{3}{\mu}\cdot\frac{m_1}{n}
\end{aligned}$$

as $m_1 = o(n)$. If $\varphi < m_1$, the probability of a good step is at least $\frac{1}{\mu}\cdot\frac{n-\varphi}{en} \geq \frac{1}{\mu}\cdot\frac{n-m_1}{en} \geq \frac{1}{2e\mu}$ since it suffices to select y_φ and to flip exactly one out of $n-\varphi$ bits. So, the probability of a good step happening before a bad step is at least

$$\frac{1/(2e\mu)}{1/(2e\mu)+3/\mu\cdot m_1/n} = \frac{1}{1+6em_1/n} = 1-\frac{6em_1/n}{1+6em_1/n} \geq 1-\frac{6em_1}{n}.$$

The probability of increasing φ from 1 to m_1 by subsequent good steps before a bad step happens is bounded by

$$\left(1-\frac{6em_1}{n}\right)^{m_1-1} \geq 1-\frac{6em_1^2}{n} = 1-o(1)$$

since $m_1^2 = o(n)$.

The next Phase 3 starts after a good Phase 2 and ends when either the global optimum is found or the potential increases to $\varphi \geq m_2$. In case $m_1 = \mu$ the goals of Phase 2 and Phase 3 coincide and Phase 3 is empty. We consider the expected increase of the potential in one generation. Let $\Delta_\varphi = \Delta_\varphi(t) := \varphi(P_{t+1}) - \varphi(P_t)$. If the current potential is not maximal, the probability of increasing the potential by 1 can be estimated by the probability of selecting y_φ and making one out of $n-\varphi$ 1-bit mutations increasing the number of 1-bits. Define $\Delta_\varphi^+ := \Delta_\varphi \cdot \mathrm{I}(\Delta_\varphi > 0)$ and $\Delta_\varphi^- := \Delta_\varphi \cdot \mathrm{I}(\Delta_\varphi < 0)$ where $\mathrm{I}(A) = 1$ if condition A is true and 0 otherwise. Then for $\varphi < m_2$

$$E\big(\Delta_\varphi^+\big) \geq \frac{1}{\mu}\cdot\frac{n-\varphi}{en}.$$

If $x_{\varphi+j}$ is selected, in order to decrease the potential it is necessary that j out of $\varphi+j$ 0-bits flip. The probability for this event is clearly bounded by $1/(j!)$. Under the condition that at least j 0-bits flip, the expected number of flipping 0-bits among $\varphi+j$ 0-bits is

bounded by $j + \varphi/n$, the term φ/n representing the expected number of flipping 0-bits among φ 0-bits. The conditional expected decrease of the potential is then at most φ/n, leading to an unconditional expectation of

$$E\big(\Delta_\varphi^-\big) \;\le\; \frac{1}{\mu} \sum_{j=1}^{n/3-\varphi-1} \frac{1}{j!} \cdot \frac{\varphi}{n} \;\le\; \frac{1}{\mu} \cdot \frac{\varphi}{n} \sum_{j=1}^{\infty} \frac{1}{j!} \;=\; \frac{1}{\mu} \cdot \frac{(e-1)\varphi}{n}$$

using $e = \sum_{j=0}^{\infty} 1/(j!)$. Putting $E\big(\Delta_\varphi^+\big)$ and $E\big(\Delta_\varphi^-\big)$ together, along with $\varphi < m_2 \le n/6$,

$$E(\Delta_\varphi) \;\ge\; \frac{1}{\mu} \cdot \left(\frac{n-\varphi}{en} - \frac{(e-1)\varphi}{n} \right) \;=\; \frac{1}{\mu} \cdot \frac{n - (e^2 - e + 1)\varphi}{en} \;\ge\; \frac{\varepsilon}{\mu}$$

for some $\varepsilon > 0$. One conclusion is that by Lemma 1, applied to random variables $X_t := m_2 - \varphi(P_t)$, the expected time until the potential has reached its maximum value m_2 or the optimum is found beforehand is bounded by $O(n\mu)$. In other words, Phase 3 ends in expected time $O(n\mu)$.

In order to estimate the error probability in this phase, we apply Lemma 2 to the potential in the interval between $a := 0$ and $b := \sqrt[3]{n}$. Note that b cannot be chosen larger as the lemma requires the starting point to be at least b. Another obstacle is that the drift ε/μ decreases with μ. So we consider the potential only in relevant steps, defined as steps where $\Delta_\varphi \neq 0$. The arguments from our estimation of $E\big(\Delta_\varphi^+\big)$, along with $\varphi \le n/6$, yield the lower bound $\mathrm{P}(\Delta_\varphi \neq 0) \ge \frac{1}{\mu} \cdot \frac{n-\varphi}{en} \ge \frac{5}{6e\mu}$. We also need an upper bound on this probability. If $y_{\varphi-k}$ is selected for $0 \le k \le \varphi$, at least k bits have to flip in order to have a relevant step. The same holds if $x_{\varphi+1+k}$ is selected for $0 \le k \le \mu-\varphi-2$. As all these solutions are selected with probability $1/\mu$, the probability of a relevant step is bounded from above by $\mathrm{P}(\Delta_\varphi \neq 0) \le \frac{1}{\mu} \cdot 2\sum_{k=0}^{\infty} \frac{1}{k!} = \frac{2e}{\mu}$. This yields for $i < n/6$

$$E(\Delta_\varphi \mid \Delta_\varphi \neq 0) \;=\; \frac{E(\Delta_\varphi)}{\mathrm{P}(\Delta_\varphi \neq 0)} \;\ge\; \frac{\varepsilon}{2e}$$

and the first condition for Lemma 2 is fulfilled. For the second condition, it is necessary for $\Delta_\varphi = -j$, $j \in \mathbb{N}$, to select some $x_{\varphi+k}$ and to flip $k + j$ bits simultaneously. Using $(k + j)! \ge k! \cdot j!$ for $k, j \in \mathbb{N}_0$ yields

$$\mathrm{P}(\Delta_\varphi = -j) \;\le\; \frac{1}{\mu} \cdot \sum_{k=0}^{\infty} \frac{1}{(k+j)!} \;\le\; \frac{1}{\mu} \cdot \frac{1}{j!} \sum_{k=0}^{\infty} \frac{1}{k!} \;=\; \frac{e}{\mu} \cdot \frac{1}{j!}.$$

For the conditional probabilities in relevant steps this means

$$\mathrm{P}(\Delta_\varphi = -j \mid \Delta_\varphi \neq 0) \;=\; \frac{\mathrm{P}(\Delta_\varphi = -j)}{\mathrm{P}(\Delta_\varphi \neq 0)} \;\le\; \frac{6e^2}{5j!} \;\le\; 2^{-j+5}$$

and the second condition holds for $\delta := 1$ and $r := 5$. Lemma 2 shows that the probability of reaching the global optimum within $2^{c\sqrt[3]{n}}$ steps is $2^{-\Omega(\sqrt[3]{n})} = o(1)$ for an appropriate constant $c > 0$. By Markov's inequality the probability that Phase 3 is not finished after this number of steps is also $o(1)$. Concluding, Phase 3 is good with probability $1 - o(1)$.

The fourth and last phase starts after a good Phase 3 and ends when the global optimum has been found. Phase 4 therefore starts with a maximum potential of m_2. If $\mu \le n/6 + 1$ then at least $2n/3$ bits have to flip simultaneously in order to create an

accepted x-individual. The probability of this event is $n^{-\Omega(n)}$ and the claim follows. If $\mu > n/6+1$ then we apply Lemma 2 to the larger interval from $a := 0$ to $b := n/6$. In the analysis of Phase 3 we have already shown the preconditions for this larger interval, hence the algorithm needs at least $2^{\Omega(n)}$ steps with probability $o(1)$. Summing up all error probabilities proves the claim for $\mu \leq n/3$.

Finally, we argue how to deal with larger population sizes, $\mu > n/3$. We relax our condition on worst-case populations P_t to

$$P_t \supseteq \{y_0, \ldots, y_\varphi, x_{\varphi+1}, \ldots, x_{n/3}\},$$

where the remaining individuals all have a number of ones in between $n/3$ and $2n/3$. These individuals therefore have Hamming distance at least $n/6$ to all search points that determine the current potential. The probability that such a search point interferes with our previous arguments is therefore at most $1/(n/6)! = n^{-\Omega(n)}$. Reinspecting the analysis for Phases 2–4, we see that compared to the setting for $\mu \leq n/3$ the probabilities for bad and relevant steps only increase by additive terms of $n^{-\Omega(n)}$. Similarly, $E(\Delta_\varphi)$ is only decreased by $n^{-\Omega(n)}$ for $0 < \varphi < n/6$. In particular, Phase 2 remains good with probability $1 - o(1)$ and the application of Lemma 2 in Phases 3 and 4 remains possible for appropriate constants ε, δ, r. □

6 Deterministic Crowding

In the deterministic crowding mechanism offspring compete directly with their respective parents. According to Mahfoud [11], in a genetic algorithm with crossover deterministic crowding works as follows. In every generation the population is partitioned into $\mu/2$ pairs of individuals, assuming μ to be even. These pairs are then recombined and mutated. Every offspring then competes with one of its parents and may replace it if the offspring is not worse.

As we do not consider crossover, we adapt the main idea of offspring competing with their parents for a steady-state mutation-based algorithm. More precisely, in the following algorithm an offspring replaces its parent if its fitness is at least as good.

Algorithm 4 (μ+1) EA with deterministic crowding

Let $t := 0$ and initialize P_0 with μ individuals chosen uniformly at random.
repeat
  Choose $x \in P_t$ uniformly at random.
  Create y by flipping each bit in x independently with probability $1/n$.
  if $f(y) \geq f(x)$ **then** $P_{t+1} = P_t \setminus \{x\} \cup \{y\}$ **else** $P_{t+1} = P_t$.
  Let $t = t + 1$.

The algorithm closely resembles a parallel (1+1) EA since μ individuals explore the landscape independently. However, in contrast to parallel runs, interactions between the individuals may be obtained by using other operators together with mutation. Recently, the mechanism together with crossover has proved to be useful in vertex cover problems by making the difference between polynomial and exponential runtimes for some instances [15]. Here we concentrate on the capabilities of guaranteeing diversity of the mechanism by analyzing the (μ+1) EA with deterministic crowding on the TWOMAX function. For sufficiently large populations the algorithm can easily reach both local optima.

Theorem 4. *The (μ+1) EA with deterministic crowding and $\mu = \mathrm{poly}(n)$ reaches on* TWOMAX *a population consisting of only local or global optima in expected time $O(\mu n \log n)$. In that case the population contains at least one global optimum with probability at least $1-2^{-\mu}$.*

Proof. The main observation for the second statement is that the individuals of the population are *independent* due to the crowding mechanism. Due to the symmetry of TWOMAX, the i-th individual in the population reaches the global optimum with probability $1/2$. The probability that at least one individual finds the global optimum is $1-(1/2)^{\mu}$.

Let T be the random time until all the individuals have reached local or global optima on TWOMAX. It is easier to find a local optimum on TWOMAX than to find a global optimum on ONEMAX, hence we estimate $E(T)$ by the expected global optimization time on ONEMAX.

Consider the following game of balls and bins (cf. the coupon collector's theorem in [12]), where bins represent bits and balls represent 1-values. Imagine a bin for every bit in the initial population, i. e., a set of μn bins labelled with their respective bits. Place a ball in each bin if the associated bit is set to 1 in the initial population. In the following generations balls may be put into empty bins according to certain rules. The game ends when all bins contain a ball—this corresponds to a population where all bits have been set to 1 and every individual represents the optimum 1^n.

Consider a generation where some individual x is selected as parent. The probability that mutation only flips a specific 0-bit x_i into a 1-bit is $1/\mu \cdot 1/n \cdot (1-1/n)^{n-1} \geq 1/(en\mu)$. As a consequence of this mutation, the offspring replaces its parent x. Compared to the previous population, the bit x_i is being switched to value 1 and we imagine that in this case a new ball is put into the empty bin for x_i. If mutation creates an offspring y with $\textsc{OneMax}(y) = \textsc{OneMax}(x)$, y again replaces x and we imagine the bins for x being rearranged so that their occupancy matches the 1-bits in y. Note that rearranging bins of an individual, i. e., rearranging bits within an individual, does not make a difference for the algorithm as all bits are treated equally. In the case where $\textsc{OneMax}(y) > \textsc{OneMax}(x)$, we imagine that $\textsc{OneMax}(y) - \textsc{OneMax}(x)$ balls are added to arbitrary empty bins of x and afterwards the bins are again rearranged to match the offspring y.

Fix an arbitrary empty bin. The probability that it receives a ball in one generation is at least $1/(en\mu)$. The probability that the bin does not receive a ball within $t := en\mu \cdot \ln(2n\mu)$ steps is bounded by $\left(1-\frac{1}{en\mu}\right)^{en\mu\cdot\ln(2n\mu)} \leq e^{-\ln(2n\mu)} = \frac{1}{2n\mu}$. By the union bound, the probability that there is still an empty bin left in the game after t steps is at most $1/2$. In case the game has not ended after t steps, we consider another period of t steps and continue waiting for the game to end. The expected number of periods until the game ends is at most 2. Hence the expected time of the game is bounded by $E(T) \leq 2t = O(\mu n \log n)$ since $\ln(2n\mu) = O(\log n)$ follows from $\mu = \mathrm{poly}(n)$. □

7 Fitness Sharing

Fitness sharing [see, e. g., 11] derates the real fitness of an individual x by an amount that represents the similarity of x to other individuals in the population. The similarity between x and y is measured by a so-called *sharing function* $\mathrm{sh}(x,y) \in [0,1]$ where a large value corresponds to large similarities and value 0 implies no similarity. The idea is that if there are several copies of the same individual in the population, these individuals have to share their fitness. As a consequence, selection is likely to remove such clusters and to keep the individuals apart. We define the *shared fitness* of x in the

population P and the fitness $f(P)$ of the population, respectively, as

$$f(x,P) \;=\; \frac{f(x)}{\sum_{y\in P}\mathrm{sh}(x,y)} \qquad \text{and} \qquad f(P) \;=\; \sum_{x\in P} f(x,P).$$

It is common practice to use a so-called *sharing distance* σ such that individuals only share fitness if they have distance less than σ. Given some distance function d, a common formulation for the sharing function is

$$\mathrm{sh}(x,y) \;=\; \max\{0, 1-(d(x,y)/\sigma)^\alpha\}$$

where α is a positive constant that regulates the shape of the sharing function. We use the standard setting $\alpha = 1$ and, following Mahfoud [11], we set the sharing distance to $\sigma = n/2$ as this is the smallest value allowing discrimination between the two branches. As TWOMAX is a function of unitation, we allow the distance function d to depend on the number of ones: $d(x,y) := \big|\,|x|_1 - |y|_1\,\big|$. Such a strategy is known as *phenotypic sharing* [11]. Our precise sharing function is then

$$\mathrm{sh}(x,y) \;=\; \max\left\{0,\; 1-2\,\frac{\big|\,|x|_1-|y|_1\,\big|}{n}\right\}.$$

We now incorporate fitness sharing into the (μ+1) EA. Our goal is to evolve a good population, hence selection works by comparing candidates for next generation's population with respect to their $f(P)$-values. This selection strategy has already been analyzed by Sudholt [18].

Algorithm 5 (μ+1) EA with fitness sharing

Let $t := 0$ and initialize P_0 with μ individuals chosen uniformly at random.
repeat
  Choose $x \in P_t$ uniformly at random.
  Create y by flipping each bit in x independently with probability $1/n$.
  Let $P_t^* := P_t \cup \{y\}$.
  Choose $z \in P_t^*$ such that $f(P_t^* \setminus \{z\})$ is maximized.
  Let $P_{t+1} = P_t^* \setminus \{z\}$ and $t = t+1$.

Note that when evaluating $f(P_t^* \setminus \{z\})$ the shared fitness values have to be recomputed for all these populations. However, with the use of dictionaries it suffices to compute $f(y)$ and the sharing values $\mathrm{sh}(x,y)$ for $x \in P_t$ only once. In addition, fitness evaluations are often the most expensive operations in evolutionary computation, so the additional effort is negligible.

We now show that the (μ+1) EA with fitness sharing can find both optima on TWOMAX. Imagining all parents and the new offspring on a scale of $|x|_1$, the individuals with the smallest and the largest number of ones have the largest distance to all individuals in the population. Therefore, fitness sharing makes these outer individuals very attractive in terms of shared fitness, hence these individuals are taken over to the next generation. This even holds if an outer individual has the worst fitness in the population.

Lemma 3. *Consider the (μ+1) EA with fitness sharing and $\mu \geq 2$ on* TWOMAX. *Let P_t^* be the enlarged parent population at some point of time t and w.l.o.g. let $P_t^* = \{x_1, \ldots, x_{\mu+1}\}$ with $|x_1|_1 \leq |x_2|_1 \leq \cdots \leq |x_{\mu+1}|_1$. If $|x_\mu|_1 < |x_{\mu+1}|_1$ then $x_{\mu+1} \in P_{t+1}$. Also, if $|x_1|_1 < |x_2|_1$ then $x_1 \in P_{t+1}$.*

Proof. We only prove $x_{\mu+1} \in P_{t+1}$ for $|x_\mu|_1 < |x_{\mu+1}|_1$. The second claim for the case $|x_1|_1 < |x_2|_1$ follows by symmetry. Let $P^- := P_t^* \setminus \{x_{\mu+1}\}$ be a "bad" new population without $x_{\mu+1}$, contradicting our claim that $x_{\mu+1}$ remains in the population. Let $P^+ := P_t^* \setminus \{x_\mu\}$ be one "good" new population where x_μ is removed instead and let $P^\cap := P^+ \cap P^-$ contain all other individuals. We will prove $f(P^-) < f(P^+)$. This implies that the bad population P^- does not have maximal population fitness among all possible next populations $P_t^* \setminus \{z\}$ examined in the selection step, hence P^+ or another "good" population is chosen and $x_{\mu+1}$ remains in the population.

We first deal with the case $f(x_\mu) \leq f(x_{\mu+1})$. Intuitively, $x_{\mu+1}$ is better than x_μ both in terms of real fitness and in terms of sharing distance to the other individuals in $P^\cap$. More precise, we have $\mathrm{sh}(x_\mu, x) > \mathrm{sh}(x_{\mu+1}, x)$ for $x \in P^\cap$ due to the ordering of the x_i and the definition of the sharing function. Hence, $\sum_{y\in P^-} \mathrm{sh}(x,y) > \sum_{y\in P^+} \mathrm{sh}(x,y)$ for $x \in P^\cap$ and $\sum_{y\in P^-} \mathrm{sh}(x_\mu, y) > \sum_{y\in P^+} \mathrm{sh}(x_{\mu+1}, y)$. Together, we obtain

$$\begin{aligned} f(P^-) &= \sum_{x\in P^\cap} \frac{f(x)}{\sum_{y\in P^-} \mathrm{sh}(x,y)} + \frac{f(x_\mu)}{\sum_{y\in P^-} \mathrm{sh}(x_\mu, y)} \\ &< \sum_{x\in P^\cap} \frac{f(x)}{\sum_{y\in P^+} \mathrm{sh}(x,y)} + \frac{f(x_{\mu+1})}{\sum_{y\in P^+} \mathrm{sh}(x_{\mu+1}, y)} = f(P^+). \end{aligned}$$

It remains to show $f(P^-) < f(P^+)$ for the case $f(x_\mu) > f(x_{\mu+1})$. This means that x_μ is better than $x_{\mu+1}$ in terms of real fitness, but worse with respect to the shared distance to all individuals in $P^\cap$. We want to examine the impact of the distance between the two individuals on the fitness and the sharing function. Therefore, we define

$$d := \min\left\{|x_{\mu+1}|_1, \frac{n}{2}\right\} - |x_\mu|_1 .$$

Note that $d \geq 0$ as $|x_\mu|_1 < n/2$ follows from $f(x_\mu) > f(x_{\mu+1})$. The minimum in the definition of d implies the following inequality.

$$\forall\, x \in P^\cap\colon\ \mathrm{sh}(x, x_{\mu+1}) \leq \mathrm{sh}(x, x_\mu) - \frac{2d}{n}, \tag{1}$$

which is immediate from the definition of $\mathrm{sh}(\cdot,\cdot)$ if $x_{\mu+1}$ is within sharing distance from x. Otherwise, it follows from

$$\mathrm{sh}(x, x_{\mu+1}) = 0 \leq \mathrm{sh}(x, x_\mu) - \frac{2(\frac{n}{2} - |x_\mu|_1)}{n} = \mathrm{sh}(x, x_\mu) - \frac{2d}{n}.$$

Inequality (1) can now be used to relate the shared fitness $f(x, P^-)$ to the shared fitness $f(x, P^+)$ for $x \in P^\cap$. This reflects the gain in total shared fitness for the individuals in $P^\cap$ if P^+ is selected instead of P^-. Using inequality (1) and the fact that the sharing function is at most 1, for all $x \in P^\cap$:

$$\begin{aligned} \frac{f(x,P^-)}{f(x,P^+)} = \frac{\sum_{y\in P^+} \mathrm{sh}(x,y)}{\sum_{y\in P^-} \mathrm{sh}(x,y)} &\leq \frac{\sum_{y\in P^\cap} \mathrm{sh}(x,y) + \mathrm{sh}(x,x_\mu) - 2d/n}{\sum_{y\in P^\cap} \mathrm{sh}(x,y) + \mathrm{sh}(x,x_\mu)} \\ &\leq \frac{\mu - 2d/n}{\mu} = 1 - \frac{2d}{n\mu}. \end{aligned} \tag{2}$$

As $f(x_{\mu+1}) < f(x_\mu)$, the total real fitness of the individuals is worse for P^+ than for P^-. We also relate the shared fitness $f(x_\mu, P^-)$ to $f(x_{\mu+1}, P^+)$ in order to estimate the loss of

shared fitness. By definition of d and TWOMAX it is easy to see that $f(x_\mu)-d \leq f(x_{\mu+1})$. Along with $f(x) \geq n/2$ for every $x \in \{0,1\}^n$, we have

$$\frac{f(x_{\mu+1})}{f(x_\mu)} \;\geq\; \frac{f(x_\mu)-d}{f(x_\mu)} \;\geq\; \frac{n/2-d}{n/2} \;=\; 1-\frac{2d}{n}.$$

Also, using inequality (1), we obtain

$$\begin{aligned}\frac{\sum_{y\in P^+}\mathrm{sh}(x_{\mu+1},y)}{\sum_{y\in P^-}\mathrm{sh}(x_\mu,y)} &= \frac{1+\sum_{y\in P^\cap}\mathrm{sh}(x_{\mu+1},y)}{1+\sum_{y\in P^\cap}\mathrm{sh}(x_\mu,y)}\\ &\leq \frac{1+\sum_{y\in P^\cap}(\mathrm{sh}(x_\mu,y)-2d/n)}{1+\sum_{y\in P^\cap}\mathrm{sh}(x_\mu,y)} \;\leq\; \frac{1+(\mu-1)(1-2d/n)}{\mu}.\end{aligned}$$

Taking the last two estimations together,

$$\begin{aligned}\frac{f(x_\mu,P^-)}{f(x_{\mu+1},P^+)} &= \frac{\sum_{y\in P^+}\mathrm{sh}(x_{\mu+1},y)}{\sum_{y\in P^-}\mathrm{sh}(x_\mu,y)}\cdot\frac{f(x_\mu)}{f(x_{\mu+1})}\\ &\leq \frac{1+(\mu-1)(1-2d/n)}{\mu(1-2d/n)} \;=\; \frac{1-\frac{2d}{n}+\frac{2d}{n\mu}}{1-\frac{2d}{n}} \;<\; 1+\frac{2d}{n\mu}. \qquad (3)\end{aligned}$$

So, when comparing P^- with P^+, we have a gain of shared fitness for all $x \in P^\cap$ and a loss of shared fitness for the remaining individual when exchanging x_μ for $x_{\mu+1}$. Putting inequalities (2) and (3) together yields

$$\begin{aligned}f(P^-) &= \sum_{x\in P^\cap} f(x,P^-)+f(x_\mu,P^-)\\ &< \sum_{x\in P^\cap} f(x,P^+)\cdot\left(1-\frac{2d}{n\mu}\right)+f(x_{\mu+1},P^+)\cdot\left(1+\frac{2d}{n\mu}\right)\\ &\leq f(P^+)-\frac{2d}{n\mu}\left(\sum_{x\in P^\cap} f(x,P^+)-f(x_{\mu+1},P^+)\right).\end{aligned}$$

Now $f(P^-) < f(P^+)$ follows if the term in parentheses is non-negative, i. e.,

$$\sum_{x\in P^\cap} f(x,P^+) \;\geq\; f(x_{\mu+1},P^+). \qquad (4)$$

Recall $|x_\mu|_1 \leq n/2$ and consider the left-hand side of inequality (4). This term is minimized if all $x \in P^\cap$ equal $x_{\mu-1}$ since then for all individuals in $P^\cap$ fitness is minimized and sharing is maximized. Along with $f(x_{\mu-1}) > f(x_{\mu+1})$ and $\mu-1 \geq 1$,

$$\begin{aligned}\sum_{x\in P^\cap} f(x,P^+) &\geq \frac{(\mu-1)\cdot f(x_{\mu-1})}{\mu-1+\mathrm{sh}(x_{\mu-1},x_{\mu+1})}\\ &> \frac{f(x_{\mu+1})}{1+\mathrm{sh}(x_{\mu-1},x_{\mu+1})} \;\geq\; \frac{f(x_{\mu+1})}{\sum_{y\in P^+}\mathrm{sh}(x_{\mu+1},y)} \;=\; f(x_{\mu+1},P^+). \quad \square\end{aligned}$$

Now it is easy to prove an upper bound on TWOMAX. To the best of our knowledge, the following theorem provides the first runtime analysis of an EA with fitness sharing for population sizes greater than 2.

Theorem 5. *The (μ+1) EA with fitness sharing and $\mu \geq 2$ reaches on* TWOMAX *a population containing both optima in expected optimization time $O(\mu n \log n)$.*

Proof. For a population P, we consider the following characteristic values as potential functions: $m_0(P)$ denotes the maximum number of zeros and $m_1(P)$ the maximum number of ones for the individuals in P. We are interested in the expected time until both potentials become n.

According to Lemma 3, both potentials cannot decrease. If $m_0(P) = k$ then we wait for an individual with k zeros to be chosen and for the number of zeros to be increased. The expected time for this to happen is bounded from above by $O(\mu \cdot n/(n-k))$. Hence, the expected time until the m_0-potential reaches its maximum value n is $O(\mu n \log n)$. A symmetrical statement holds for the m_1-potential, hence the expected time until both optima are found is bounded by $O(\mu n \log n)$. □

8 Experiments

Our negative results for the (μ+1) EA without diversification and the genotype diversity mechanism only hold for relatively small population sizes. We believe that the same results also hold for larger values of μ, but a theoretical analysis is challenging. Initial experiments have shown that also with larger μ extinction is still likely, but the questions concerning which branch gets extinct and when extinction happens are determined by long-term dynamics that are very difficult to handle analytically.

A typical behavior is that one branch starts lagging behind a little. Then chances to create offspring on higher fitness levels are lower for this branch, while the other branch has an advantage in this respect. When climbing up the next fitness levels, this effect may intensify until the branch that is behind gets extinct. However, it may also happen that by chance the branch that is behind creates an offspring on a good fitness level ℓ and then the branch is safe from extinction until level ℓ becomes the worst fitness level in the population. This gives the branch some time to recover, which makes it hard to predict when a branch will get extinct. It is even hard to tell which branch is "behind". One branch may consist of few good individuals and the other one of many inferior solutions. Which one is more likely to survive in the long run? We feel that completely new methods have to be developed in order to understand these long-term dynamics of the processes.

We rely on experiments to find out how large a population has to be to avoid extinction. This also allows a more detailed comparison of diversity-preserving mechanisms. We consider exponentially increasing population sizes $\mu = 2, 4, 8, \ldots, 1024$ for $n = 30$ and perform 100 runs in each setting. An obvious performance measure is the number of runs where the global optimum is found. However, for hill climbers like the (1+1) EA this measure fluctuates around 50 runs. In order to obtain a more clear picture without this random fluctuation, we instead consider the number of runs where both 0^n and 1^n were present in the population at the same time. Such a run is called *successful* hereinafter. In all non-successful runs we have a conditional probability of exactly $1/2$ that the global optimum was found due to the symmetry of TWOMAX, provided $\mu \geq 2$. Moreover, we record the maximum progress on the branch that gets extinct, computed as $\min\{m_0(P_t),\ m_1(P_t)\}$, where $m_0(P_t)$ ($m_1(P_t)$) denotes the maximum number of zeros (ones) in the population P_t at time t. In case both 0^n and 1^n are contained in the population, the maximum progress equals n. For fitness sharing we present our theoretical results as the outcome of experiments is predetermined. The number of successful runs is clearly 100 and the maximum progress is $n = 30$.

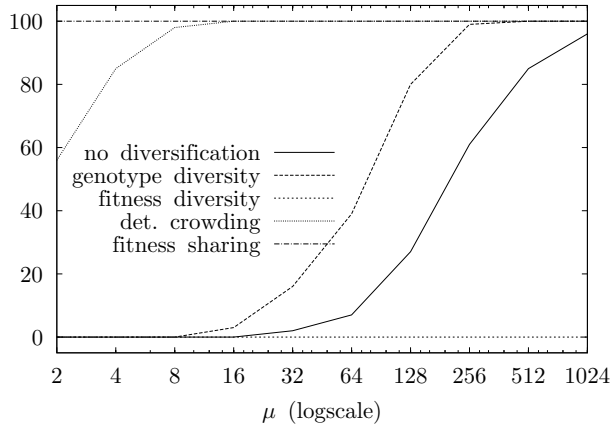

Figure 2: The number of successful runs among 100 runs for $n = 30$ and $\mu = 2, 4, 8, \ldots, 1024$. A run is called successful if 0^n and 1^n were present in the population at the same time.

The choice of the stopping criterion is non-trivial. A natural design choice is to stop the algorithm after convergence to local or global optima. For fitness sharing, deterministic crowding, and the (μ+1) EA without diversification, we stop the run when either the run is successful or when the whole population only consists of copies of 0^n or 1^n. For genotype diversity and, especially, fitness diversity there is no such convergence as copies of 0^n and 1^n are not allowed. The genotype diversity mechanism gets stuck when all individuals are as close to 0^n or 1^n as possible. The same happens for fitness diversity when $\mu \ll n/2$ and all individuals are stacked one after another on one branch. However, the fitness diversity mechanism for $\mu > n/2$ does not converge at all because the population performs a random walk with a drift pointing away from the optimum when it gets close enough. Therefore, we stop a run for these two mechanisms after 3600μ generations. The time bound 3600μ was chosen for the following reason. For $n = 30$ we have $360\mu > \mu \cdot en(\ln n + 1)$. This term is the upper bound for the expected time to reach either 0^n or 1^n mentioned in Section 2. This bound holds for all algorithms presented in this work. Moreover, the random time is concentrated around the expectation as can be seen from the proof of Theorem 4. Our bound 3600μ is more than ten times larger than the expectation. This is enough time for the two algorithms to converge or to reach a meaningful equilibrium state.

Figure 2 shows the number of successful runs. While fitness sharing is always successful, fitness diversity was never found to be successful. Using deterministic crowding, the success probability increases very steeply compared to the scenarios of no diversification and genotype diversity. Although genotype diversity is a rather weak mechanism, it turns out to be more successful than no diversification.

Figure 3 shows the average maximum progress in 100 runs. It is obvious that fitness sharing and deterministic crowding perform well due to their high success probabilities. But here also fitness diversity has an effect as its progress indicator increases with increasing μ, although more slowly than the indicator of all the other algorithms. Concerning the (μ+1) EA without diversification and with genotype diversity, we can see from Figure 3 that for small population sizes extinction occurs very early and for low fitness values. Contrarily, our theoretical arguments were based on the very last fit-

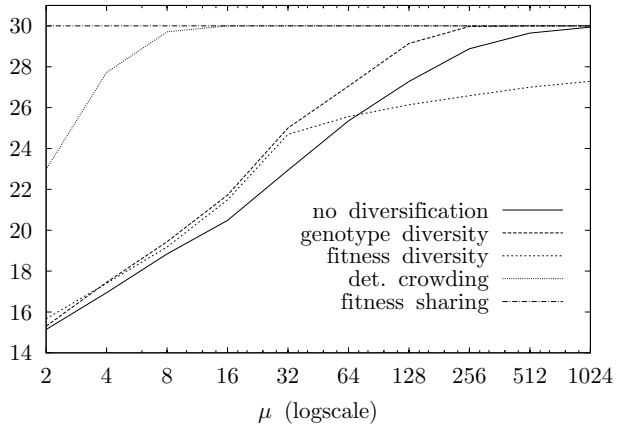

Figure 3: The average maximum progress on the branch that is behind, measured among 100 runs, for $n = 30$ and $\mu = 2, 4, 8, \ldots, 1024$.

ness levels where our estimates of the extinction probabilities were best. This strengthens our impression that extinction is due to much more complex long-term effects than used in our proofs.

9 Conclusions

We have examined the behavior of different diversity-preserving mechanisms on a fitness landscape consisting of two hills with symmetric slopes. We rigorously proved that without any diversification the whole population of the (μ+1) EA runs into the local optimum with probability almost $1/2$ (Theorem 1). This still holds if we avoid genotype duplicates or fitness duplicates. An implication is that for these algorithms the population is nearly useless as we experience the same performance as for simple hill climbers like local search or the (1+1) EA.

On the other hand, stronger diversity-preserving mechanisms like fitness sharing and deterministic crowding allow the (μ+1) EA to find both optima of our test function TWOMAX with high probability. Deterministic crowding performs as well as independent runs and the probability of not finding both optima decreases exponentially with μ. Fitness sharing using a phenotypic distance function always finds both optima efficiently for arbitrary populations of size $\mu \geq 2$.

Our theoretical results and the experiments from Section 8 have also revealed important open problems. Theorems 1 and 2 apply only to sublinear population sizes. Our experimental results indicate a similar behavior also for larger populations, but a theoretical analysis is difficult. It seems that a more thorough understanding of the long-term dynamics of genetic drift is required to strengthen the theorems.

References

[1] N. Chaiyaratana, T. Piroonratana, and N. Sangkawelert. Effects of diversity control in single-objective and multi-objective genetic algorithms. *Journal of Heuristics*, 13 (1):1–34, 2007.

[2] S. Fischer and I. Wegener. The one-dimensional Ising model: Mutation versus recombination. *Theoretical Computer Science*, 344(2–3):208–225, 2005.

[3] T. Friedrich, N. Hebbinghaus, and F. Neumann. Rigorous analyses of simple diversity mechanisms. In *Proc. of the annual Conference on Genetic and Evolutionary Computation (GECCO '07)*, pages 1219–1225. ACM Press, 2007.

[4] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt. Theoretical analysis of diversity mechanisms for global exploration. In *Proc. of the annual Conference on Genetic and Evolutionary Computation (GECCO '08)*, pages 945–952. ACM Press, 2008.

[5] D. E. Goldberg, C. Van Hoyweghen, and B. Naudts. From twomax to the Ising model: Easy and hard symmetrical problems. In *Proc. of the annual Conference on Genetic and Evolutionary Computation (GECCO '02)*, pages 626–633. Morgan Kaufmann, 2002.

[6] B. Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 14:502–525, 1982.

[7] J. He and X. Yao. A study of drift analysis for estimating computation time of evolutionary algorithms. *Natural Computing*, 3(1):21–35, 2004.

[8] M. Hutter and S. Legg. Fitness uniform optimization. *IEEE Transactions on Evolutionary Computation*, 10:568–589, 2006.

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

[10] T. Jansen and I. Wegener. Real royal road functions: where crossover provably is essential. *Discrete Applied Mathematics*, 149(1-3):111–125, 2005.

[11] S. W. Mahfoud. Niching methods. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*, pages C6.1:1–4. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

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

[13] P. S. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. In *Proc. of the International Conference on Parallel Problem Solving From Nature (PPSN X)*, volume 5199 of *LNCS*, pages 82–91. Springer, 2008.

[14] P. S. Oliveto, J. He, and X. Yao. Computational complexity analysis of evolutionary algorithms for combinatorial optimization: A decade of results. *International Journal of Automation and Computing*, 4(3):281–293, 2007.

[15] P. S. Oliveto, J. He, and X. Yao. Population-based evolutionary algorithms for the vertex cover problem. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC '08)*, pages 1563–1570, 2008.

[16] M. Pelikan and D. E. Goldberg. Genetic algorithms, clustering, and the breaking of symmetry. In *Proc. of the International Conference on Parallel Problem Solving From Nature (PPSN VI)*, pages 385–394. Springer, 2000.

[17] T. Storch and I. Wegener. Real royal road functions for constant population size. *Theoretical Computer Science*, 320:123–134, 2004.

[18] D. Sudholt. Crossover is provably essential for the Ising model on trees. In *Proc. of the annual Conference on Genetic and Evolutionary Computation (GECCO '05)*, pages

1161–1167. ACM Press, 2005.

[19] D. Sudholt. Memetic algorithms with variable-depth search to overcome local optima. In *Proc. of the annual Conference on Genetic and Evolutionary Computation (GECCO '08)*, pages 787–794. ACM Press, 2008.

[20] R. K. Ursem. Diversity-guided evolutionary algorithms. In *Proc. of the International Conference on Parallel Problem Solving From Nature (PPSN VII)*, pages 462–471. Springer, 2002.

[21] I. Wegener and C. Witt. On the optimization of monotone polynomials by simple randomized search heuristics. *Combinatorics, Probability and Computing*, 14:225–247, 2005.

[22] C. Witt. Runtime analysis of the (μ+1) EA on simple pseudo-Boolean functions. *Evolutionary Computation*, 14(1):65–86, 2006.