Escaping Local Optima with Diversity Mechanisms and Crossover

Duc-Cuong Dang
University of Nottingham
Nottingham, UK

Tobias Friedrich
Hasso Plattner Institute
Potsdam, Germany

Timo Kötzing
Hasso Plattner Institute
Potsdam, Germany

Martin S. Krejca
Hasso Plattner Institute
Potsdam, Germany

Per Kristian Lehre
University of Nottingham
Nottingham, UK

Pietro S. Oliveto
University of Sheffield
Sheffield, UK

Dirk Sudholt
University of Sheffield
Sheffield, UK

Andrew M. Sutton
Hasso Plattner Institute
Potsdam, Germany

ABSTRACT

Population diversity is essential for the effective use of any crossover operator. We compare seven commonly used diversity mechanisms and prove rigorous run time bounds for the (μ+1) GA using uniform crossover on the fitness function Jump_k. All previous results in this context only hold for unrealistically low crossover probability $p_c = O(k/n)$, while we give analyses for the setting of constant $p_c < 1$ in all but one case. Our bounds show a dependence on the problem size n, the jump length k, the population size μ, and the crossover probability p_c. For the typical case of constant $k > 2$ and constant p_c, we can compare the resulting expected optimisation times for different diversity mechanisms assuming an optimal choice of μ:

- $O(n^{k-1})$ for duplicate elimination/minimisation,
- $O(n^2 \log n)$ for maximising the convex hull,
- $O(n \log n)$ for det. crowding (assuming $p_c = k/n$),
- $O(n \log n)$ for maximising the Hamming distance,
- $O(n \log n)$ for fitness sharing,
- $O(n \log n)$ for the single-receiver island model.

This proves a sizeable advantage of all variants of the (μ+1) GA compared to the (1+1) EA, which requires $\Theta(n^k)$. In a short empirical study we confirm that the asymptotic differences can also be observed experimentally.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

Keywords

Genetic algorithms; recombination; crossover; diversity; run time analysis; theory

1. INTRODUCTION

There are numerous empirical and a few theoretical studies supporting the claim *that* crossover is beneficial. However, we are still far from a good understanding *why* crossover is beneficial. This is not only true for evolutionary computation, but also for models in population genetics. When and how recombination increases the speed of adaption is a fundamental question. It is also called "the queen of problems in evolutionary biology" by Graham Bell [3] and others.

Understanding *why* recombination improves evolution is a major open problem in both evolutionary computation and population genetics. The key to a successful crossover step is a sufficient amount of *diversity* within the population. A diverse population has higher chances to find dissimilar individuals and to create fitter offspring by recombining different building blocks. Diversity-preserving mechanisms can therefore significantly enhance the performance of crossover.

Most studies on diversity-preserving mechanisms are solely empirical [5, 15], for genetic programming [1, 4], or on the infinite population model [2]. We focus on the simple genetic algorithm with finite population called (μ+1) GA. Our aim is to develop rigorous run time bounds of the (μ+1) GA with different diversity mechanisms. We want to study how diversity helps to escape local optima. The standard testbed for this is the Jump_k landscape. $\text{Jump}_k \colon \{0,1\}^n \to \mathbb{N}$ is a simple pseudo-Boolean test function with a large plateau for x with $|x|_1 = n - k$ and a unique global maximum at $|x|_1 = n$. The name is derived from the fact that a (1+1) EA has to *jump* over a gap of Hamming distance k from the plateau to the global optimum. For a formal introduction of the (μ+1) GA and Jump_k, see Section 2.

In this paper, we study rigorously a variety of diversity mechanisms of the (μ+1) GA on the Jump_k function. We observe on the Jump_k function how a diverse population spreads on local optima and eventually manages to escape them. Our results show that already small changes in the tie-breaking rule of the (μ+1) GA can make a big difference. We rigorously prove asymptotic run time upper bounds for different diversity mechanism. The respective proofs itself give insight into *how* the diversity mechanism help escaping local optima.

Run time results. The first rigorous analysis of Jump_k was achieved by Jansen and Wegener [11]. They proved that the expected optimisation time of the (1+1) EA on Jump_k is $\Theta(n^k + n \log n)$ [11, Thm. 4]. Jansen and Wegener [11] and Kötzing, Sudholt, and Theile [12] also analyzed the (μ+1) GA, but both their analyses require a vanishing crossover probability of order $p_c = O(k/n)$. To the best of our knowledge, this is the first rigorous analysis of the (μ+1) GA on Jump_k that shows an advantage of crossover

GECCO '16 July 20-24, 2016, Denver, CO, USA

ACM ISBN 978-1-4503-4206-3/16/07.

DOI: http://dx.doi.org/10.1145/2908812.2908956

Mechanism	General case	$k = 2$	$k = 4$
Duplicate elimination, Theorem 3.1	$\mathrm{O}\big(n \log n + n^{k-1}\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}\big(n^3\big)$
Duplicate minimisation, Theorem 3.2	$\mathrm{O}\big(n \log n + n^{k-1}\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}\big(n^3\big)$
Deterministic crowding, Theorem 3.3	$\mathrm{O}\big(n \log n + n\mathrm{e}^{5k}\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}(n \log n)$
Convex hull maximisation, Theorem 4.1	$\mathrm{O}\big(n^2 \log n + 4^k\big)$	$\mathrm{O}\big(n^2 \log n\big)$	$\mathrm{O}\big(n^2 \log n\big)$
Hamming distance max., Theorem 4.2	$\mathrm{O}\big(n \log n + nk \log k + 4^k\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}(n \log n)$
Fitness sharing, Theorem 4.4	$\mathrm{O}\big(n \log n + nk \log k + 4^k\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}(n \log n)$
Island model, Theorem 5.1	$\mathrm{O}\big(n \log n + kn + 4^k\big)$	$\mathrm{O}(n \log n)$	$\mathrm{O}(n \log n)$

Table 1: Comparison of the asymptotically best known run times bounds for Jump_k with $k = \mathrm{O}(1)$. All results hold for a constant crossover probability $p_\mathrm{c} < 1$; expect deterministic crowding, which only holds for $p_\mathrm{c} = k/n$. For all diversity mechanisms, we assume a best-possible choice of the population size μ, which is $\mu = 2$ in all cases. More general results are presented in the respective theorem.

for a crossover probability up to constant $p_\mathrm{c} < 1$. All of our results (except deterministic crowding) still hold for constant crossover probabilities.

Our results are summarised in Table 1. The table presents the run time bound for a best-possible population size μ for each diversity mechanism. Our main results are as follows.

We consider three varieties of diversity-preserving mechanisms: (1) methods that encourage *genotypic uniqueness*, (2) methods that increase *genotypic distance*, and (3) methods that employ parallelism. In the first variety, tie-breaking mechanisms are used to promote genotypic uniqueness in the population. The method of **duplicate elimination** breaks ties in the selection process by making sure an offspring replaces a duplicated string out of the least-fit individuals. The advantage is that the crossover operator does not have to wait to get two different individuals. Theorem 3.1 shows that for $\mu, k = \mathrm{o}(n)$ this gives an expected optimisation time of $\mathrm{O}\big(\mu^2 n + n \log n + n^{k-1}\big)$.

Very similar to duplicate elimination is **duplicate minimisation.** In this case ties are broken such that the least-fit individual that has the highest number of duplicates is removed. This lowers the time to get many pairs of different individuals. For $\mu, k = \mathrm{o}(n)$ this results in Theorem 3.2 to the slightly better expected optimisation time of $\mathrm{O}\big(\mu n + n \log n + \mu \log \mu + n^{k-1}\big)$.

Deterministic crowding always chooses a parent of the current offspring for removal in the selection. This allows to get different individuals more efficiently. Theorem 3.3 shows that in this case for $p_\mathrm{c} = k/n$, $k = \mathrm{o}(\sqrt{n})$ the expected optimisation time is $\mathrm{O}\big(\mu n \log n + n\mathrm{e}^{5k}\mu^{k+2}\big)$. Note that our result for deterministic crowding is the only analysis which does not extend up to $p_\mathrm{c} = \Theta(1)$.

The second variety of diversity-preserving mechanisms we investigate are based on increasing genotypic distance between individuals in the population. A geometric crossover tends to reduce the size of the convex hull of the population [13]. It is therefore natural to break ties via a **Convex hull maximisation.** For constant $p_\mathrm{c} < 1$, $\mu \leq n/k$, $k \leq n/2$, Theorem 4.1 shows an expected optimisation time of $\mathrm{O}\big(\mu n^2 \log n + 4^k/p_\mathrm{c}\big)$.

Maximising the convex hull can be simulated by **Maximising the total Hamming distance** among the population. For constant $p_\mathrm{c} < 1$, $\mu < n/(2k)$, $k \leq n/8$, Theorem 4.2 shows an expected optimisation time of $\mathrm{O}(\mu n \log n + \mu^2 kn \log(\mu k) + 4^k/p_\mathrm{c})$. Theorem 4.4 proves that the same bound holds for **Fitness sharing,** where the fitness of an individual is lowered if other individuals are close-by.

The third and final variety of mechanism we study is based on relying on parallelism to obtain and preserve diversity. In an **island model,** diversity is attained by keeping parts of the population on separate processors. For the single-receiver island model [16], for $p_\mathrm{c} = 1$, $\mu \geq 2$, $k = \mathrm{o}(\sqrt{n}/\mu)$, Theorem 5.1 shows an expected optimisation time of $\mathrm{O}\big(n \log n + \mu^2 kn + \mu^2 4^k\big)$.

Experiments. Note that all results obtained are asymptotic upper bounds and not necessarily tight. Thus, we complement our theoretical results with a small experimental study of Jump_k. We observe that already for small k, the (μ+1) EA requires an enormous time to find the global optimum; while the (μ+1) GA is much faster already without any diversity mechanism. The different diversity mechanisms of the (μ+1) GA are all beneficial. We can empirically observe a ranking of which diversity mechanism helps the most to speed up Jump_k.

2. PRELIMINARIES

Test function. We are concerned with optimising the pseudo-Boolean fitness functions $\mathrm{Jump}_k \colon \{0,1\}^n \to \mathbb{N}$. Jump_k was introduced by Jansen and Wegener [11] and is defined, for all $x \in \{0,1\}^n$, as

$$\mathrm{Jump}_k(x) = \begin{cases} k + |x|_1 & \text{if } |x|_1 = n \vee |x|_1 \leq n - k\,, \\ n - |x|_1 & \text{otherwise,} \end{cases}$$

where $|x|_1 = \sum_{i=1}^{n} x_i$ is the number of 1-bits in x. Jump_k is symmetric, i.e., it only depends on the number of 1s and not their position. It is locally optimal for any individual x with $|x|_1 = n - k$, the *plateau*, however, its only global maximum is the all-ones string 1^n. Between the plateau and the global optimum there is a gap of Hamming distance k, which has to be *jumped over* in order to optimize the function, thus the name.

Genetic algorithm. Our algorithm of interest is the (μ+1) GA. It makes use of a mutation operator 'mutate' and a binary crossover operator 'crossover'. mutate(x) takes an individual x and flips each bit independently with probability $1/n$. crossover(x, y) performs a *uniform crossover* of x and y, i.e., each bit from the resulting string is a uniformly at random chosen bit from the parents' respective bits. Crossover is only performed with probability p_c in each iteration, mutation is always performed (after crossover, if any crossover is performed). In contrast to previous analyses of the (μ+1) GA, we also allow crossover probabilities $p_\mathrm{c} = 1 - \Omega(1)$.

We always remove elements with lowest fitness. However, we do not necessarily break ties uniformly at random but according to specific tie-breaking rules determined by the applied diversity mechanism. The pseudocode for the (μ+1) GA can be seen in Algorithm 1.

Algorithm 1: The (μ+1) GA for given μ, p_c

```
1  P ← μ individuals, uniformly at random from {0,1}^n;
2  while 1^n ∉ P do
3      Choose p ∈ [0,1] uniformly at random;
4      if p ≤ p_c then
5          Choose x, y ∈ P uniformly at random;
6          z ← mutate(crossover(x, y));
7      else
8          Choose x ∈ P uniformly at random;
9          z ← mutate(x);
10     P ← P ∪ {z};
11     Choose one element from P with lowest fitness,
       breaking ties according to the tie-breaking rule, and
       remove it from P;
```

The most interesting behaviour of the diversity mechanisms presented in this paper occurs after the entire population is stuck at local optima. However, for completeness, we also want to bound the time it takes for the population to get into such a state. The following lemma is easily proved in the same manner as for the (μ+1) EA [17] by noting that (1) before the population reaches the plateau, Jump_k is identical to OneMax and (2) the (μ+1) GA performs the same steps as the (μ+1) EA during generations that it does not perform crossover. Therefore, any slow-down caused by crossover can only contribute a constant factor to the run time, as long as $p_c = 1-\Omega(1)$. We only need an extra $\mu \log \mu$ term that captures the waiting time until all individuals are on the plateau.

Lemma 2.1. *Consider the (μ+1) GA optimizing* Jump_k *with* $p_c = 1 - \Omega(1)$. *Then the expected time until the entire population is on the plateau is* $O(\mu n + n \log n + \mu \log \mu)$.

After sufficient progress is made in diversity, crossover and mutation can work together to create an optimal solution in $o(n^k)$ time. This is captured by the following lemma and will be necessary for some of the results later in the paper.

Lemma 2.2. *Consider a population P on the* Jump_k *plateau ($f(x) = n - k$ for all $x \in P$). We partition P into equivalence classes. For any constant $0 < c < 1$, if the largest equivalence class has size at most $c\mu$, then the optimal solution is created by uniform crossover followed by mutation with probability $\Omega(n^{-(k-1)})$.*

Proof. Since the size of the largest equivalence class is no larger than $c\mu$, the probability that two distinct parents are selected for crossover is $\Omega(1)$. For the remainder of the proof, we assume that two parents x, y are selected with $x \neq y$.

Let $2d > 0$ denote the Hamming distance between x and y. Then x and y have d 1s among the $2d$ bits that differ between parents and $n-k-d$ 1s outside this area. Assume that crossover sets exactly i out of these $2d$ bits to 1, which happens with probability $\binom{2d}{i}2^{-2d}$. Then mutation needs to flip the remaining $k+d-i$ 0s to 1. The probability of this occurring is

$$\sum_{i=0}^{2d} \binom{2d}{i} \frac{1}{2^{2d} n^{k+d-i}} \left(1-\frac{1}{n}\right)^{n-k-d+i} = \Omega(n^{-(k-1)}) ,$$

where we bound the sum by dropping all but the zeroth term and use $4^{-d} \geq \frac{1}{4} n^{-(d-1)}$, since $d > 0$ and we take n to be large enough. □

3. DIVERSITY WITH UNIQUENESS

We first consider diversity mechanisms that correspond to tie-breaking rules that try to promote having unique individuals in the population either by explicitly eliminating duplicates or by enforcing selection to occur between parent and offspring (who are more likely to be non-unique).

3.1 Duplicate Elimination

In this setting, we consider the (μ+1) GA using *duplicate elimination* as a tie-breaking mechanism that operates as follows. When breaking ties on the lowest-fitness individual in line 11 of Algorithm 1, if there are no duplicates among the least-fit individuals, one is chosen at random to remove. Otherwise, we always choose one so that the number of duplicated strings decreases.

Theorem 3.1. *Consider the (μ+1) GA using duplicate elimination to break ties, a crossover probability such that $p_c = \Omega(1)$ and $p_c = 1 - \Omega(1)$, and a population size of $\mu < k(n-k)/2$. The expected optimisation time on* Jump_k *with $k = o(n)$ is $O(\mu^2 n + n \log n + n^{k-1})$.*

Proof. By Lemma 2.1 and our upper bound on μ, the expected time until the whole population is on the plateau is $O(\mu n + n \log n)$. Let $0 < c < 1$ be an arbitrary constant. We argue that after $O(\mu^2 n)$ generations in expectation, there are at most $c\mu$ duplicates in the population. A *duplicate pair* is a pair (x, y) such that $x = y$. For all $z \in \{0,1\}^n$ we define the (possibly empty) set $S_2(z) = \{y \in \{0,1\}^n : f(y) = n - k \wedge d_H(z, y) = 2\}$ where d_H denotes Hamming distance. Let $x \in P$ be conditioned on the event that every point in P is contained on the plateau. Then $|S_2(x)| = k(n-k)$, so due to our bounds on μ, $|S_2(x) \setminus P| \geq k(n-k)/2$.

If there are duplicates in the population, a new plateau point can be generated as follows. First, select a duplicate pair x, x' as parents and perform crossover to obtain x''. Obviously $x = x' = x''$. Now, mutation flips exactly two specific bits of x'' to create one of the points in $S_2(x'') \setminus P$. We call such a mutation a *novel* mutation. The probability for any novel mutation on x'' is at least

$$\frac{1}{n^2}\left(1-\frac{1}{n}\right)^{n-2} |S_2(x'') \setminus P| \geq \frac{1}{n^2}\frac{k(n-k)}{2e} = \Omega(1/n) .$$

As long as there are at least $c\mu$ duplicates, the probability of choosing a duplicate pair for parents can be bounded as follows. Consider a partition of the population at time t such that each partition contains all copies of a particular string. Let $s_1 \geq s_2 \geq \cdots \geq s_a \geq 1$ denote the sequence of partition sizes. Let $\ell^* = \max\{i : s_i > 1\}$. If $\sum_{i=1}^{\ell^*} s_i \geq c\mu$, then the number of duplicate pairs is at least

$$\sum_{i=1}^{\ell^*} \binom{s_i}{2} \geq \frac{1}{2}\sum_{i=1}^{\ell^*} s_i \geq c\mu/2 .$$

Therefore, under the condition that there are at least $c\mu$ duplicates, the probability that a duplicate pair is selected for recombination is at least $(c\mu/2)/\binom{\mu}{2} = \Omega(1/\mu)$.

If a novel mutation occurs in the offspring, then it will be accepted, and consequently a duplicate will be removed from the population. The number of duplicates cannot increase. Moreover, it decreases in each generation with probability $\Omega\big(1/(\mu n)\big)$. Hence, the expected waiting time until all but $c\mu$ duplicates have been removed from the plateau is $\mathrm{O}(\mu^2 n)$.

After this time, we maintain the invariant that the size of the largest equivalence class cannot be higher than $c\mu$. Thus, in each subsequent generation, the probability of generating the optimal string is bounded from below by Lemma 2.2. The expected number of generations until the optimal string appears in the population after this point is thus $\mathrm{O}(n^{k-1})$. □

3.2 Duplicate Minimisation

Duplicate minimisation is similar to duplicate elimination, except when breaking ties we do not choose an arbitrary duplicate, but the an individual that has the highest number of duplicates.

Theorem 3.2. *Consider the (μ+1) GA using duplicate minimisation to break ties, a crossover probability such that $p_c = \Omega(1)$ and $p_c = 1 - \Omega(1)$, and a population size of $\mu < k(n-k)/2$. The expected optimisation time on Jump_k with $k = \mathrm{o}(n)$ is $\mathrm{O}\big(\mu n + n \log n + n^{k-1}\big)$.*

Proof. The proof is identical to the proof of Theorem 3.1, except how we handle the initial gain of diversity on the plateau. In the case of duplicate minimisation, we have the extra property that the size of the largest equivalence class cannot increase over time. Hence, we only have to wait until the size of the largest equivalence class is at most $c\mu$ (instead of the entire duplicate count, as with duplicate elimination). This saves us an extra μ-factor in the waiting time to reach a point where we can apply Lemma 2.2.

Again, since $\mu = \mathrm{O}(n)$, Lemma 2.1 bounds the expected time until the whole population is on the plateau by $\mathrm{O}(\mu n + n \log n)$, hence, for the remainder of the proof, we assume that the entire population has already reached the plateau. Let X_t be the count of duplicates in the population at time t. Let Y_t denote the size of the largest equivalence class, that is, the cardinality of the largest equivalence class in the population at time t. Let x be the offspring generated at time t. If x belongs to one of the partitions of size Y_t, then the size of one of these partitions is temporarily increased before survival selection, but then duplicate minimisation ensures $Y_{t+1} = Y_t$ since one of the members of the partition of size $Y_t + 1$ is removed uniformly at random. In every other case, the size of the largest partition stays the same or decreases.

Thus we only must wait until $Y_t \leq c\mu$ to apply Lemma 2.2. In each iteration, X_t is decreased by one if a novel point is created. As stated in the proof of Theorem 3.1, the probability of a novel mutation given a duplicate pair for parents is $\Omega(1/n)$. Conditional on $Y_t > c\mu$, the probability of selecting a duplicate pair and subsequently creating a novel offspring is at least $\binom{Y_t}{2}/\big(n\binom{\mu}{2}\big) > \binom{c\mu}{2}/\big(n\binom{\mu}{2}\big) = \Omega(1/n)$.

Let $(\widehat{X}_t)_{t \geq 0}$ be the stochastic process defined by

$$\widehat{X}_t = \begin{cases} X_t & \text{if } Y_t > c\mu \ , \\ 0 & \text{otherwise.} \end{cases}$$

Since $\widehat{X}_t - \widehat{X}_{t+1} \geq 0$ and decreases by at least one with probability $\Omega(1/n)$, the waiting time until $Y_t \leq c\mu$ is $\mathrm{O}(\mu n)$. From this point and beyond, the size of the largest equivalence class is at most $c\mu$ and the proof is completed by applying Lemma 2.2. □

3.3 Deterministic Crowding

We consider deterministic crowding as described in [8]. In effect, this tie-breaking rule always chooses a parent individual of the current offspring for removal in the selection phase (in particular, the offspring always survives). Thus, for mutation, the parent is always removed; for crossover, one parent uniformly chosen at random is removed.

Theorem 3.3. *Consider the (μ+1) GA with $p_c = k/n$. Suppose that ties in the selection procedure are handled by deterministic crowding. Then the expected number of iterations until an optimal individual is created when running on Jump_k, $k = \mathrm{o}(\sqrt{n})$, is $\mathrm{O}(\mu n + n \log n + \mu \log \mu + n \mathrm{e}^{5k} \mu^{k+2})$.*

Proof. According to Lemma 2.1, after $\mathrm{O}(\mu n + n \log n + \mu \log \mu)$ rounds, all individuals are on the plateau.

The remaining part of this proof follows the ideas presented in the one from Theorem 7 of [12][1]. We want to make sure that we end up having two individuals that do not have a 0 in common and that these get chosen for crossover that succeeds in creating the optimum.

This process is divided into two phases. Phase 1 considers the probability of mutating two individuals such that they end up sharing no 0. Phase 2 then considers the probability of a crossover to occur that chooses the two individuals of phase 1. The phases are, too, separated into several events that are sufficient for the desired outcome.

Phase 1 lasts n rounds without any crossover and should generate two different individuals that do not share a single 0. Phase 2 goes on for n/k rounds and should generate the optimum. Therefore, a harmful mutation, i.e., one that changes at least one of the two designated individuals, must not occur.

We start off with phase 1 and decompose this phase into the events E1–E5.

E1: The event that no crossover occurs in n rounds. Its probability is $(1-p_c)^n \geq (1-k/n)^n \geq \mathrm{e}^{-k}\big(1 - \mathrm{o}(1)\big)$.

E2: The event, conditional on E1, that there are at least k mutations during phase 1 that flip two bits such that a 1-bit and a 0-bit get flipped. Note that an individual generated this way is on the plateau.

A single such mutation happens with probability $q := k/n \cdot (n-k)/n \cdot (1-1/n)^{n-2} \geq k/n \cdot 3^{-1}$ if n is large enough. Trivially, $q \leq k/n$ holds.

The probability of E2 happening is therefore at least

$$\binom{n}{k} q^k (1-q)^{n-k} \geq \frac{n^k}{k^k} \left(\frac{k}{n} \cdot \frac{1}{3}\right)^k \left(1 - \frac{k}{n}\right)^{n-k}$$

$$\geq 3^{-k} \left(1 - \frac{k}{n}\right)^{\left(\frac{n}{k}-1\right)k} \geq (3\mathrm{e})^{-k} .$$

E3: The event, conditional on E1, that any mutation that creates a new individual on the plateau does so by just

[1] In this proof, we set $p_c = k/n$ instead of bounding it only from above. This corrects an error that appears in the proof of Theorem 7 of [12], in which the reciprocal of the crossover probability erroneously does not appear in the run time bound.

flipping a single 0 (and of course a single 1), i.e., it is unlikely that at least two 0s get flipped, the probability of which would be at most $\binom{k}{2} \cdot 1/n^2 \leq k^2/n^2$.

The probability of this never happening during phase 1 and thus the probability of E3 is therefore bounded by

$$\left(1 - \frac{k^2}{n^2}\right)^n \geq 1 - \mathrm{O}\left(\frac{k^2}{n}\right) \overset{k=\mathrm{o}(\sqrt{n})}{\geq} 1 - \mathrm{o}(1) \ .$$

E4: The event, conditional on E1, E2, and E3, that two designated individuals get chosen for the k mutations. In the end, these individuals should be the ones chosen for crossover. The probability of choosing the correct individuals for mutation is at least $(2/\mu)^k$.

Our tie-breaking rule is deterministic crowding, so the offspring will always survive because the parent will be removed. Hence, the probability for E4 is at least $(2/\mu)^k$.

E5: The event, conditional on E1 through E4, that the two individuals from E4 actually drift apart such that, in the end, they do not share any of their 0s. To do so, the individuals must increase their Hamming distance to one another by 2. Let i denotes the number of 0-bits that do not have to be mutated anymore. This probability is at least

$$\begin{aligned}
&\prod_{i=1}^{k-1} \left(\frac{k-i}{k} \cdot \frac{n-k-i}{n-k}\right) \geq \left(\frac{n-2k}{n-k}\right)^{k-1} \prod_{i=1}^{k-1} \frac{k-i}{k} \\
&\quad \geq \left(1 - \frac{k}{n-k}\right)^{k-1} \cdot \frac{(k-1)!}{k^{k-1}} \\
&\quad \geq \left(1 - \frac{k^2}{n-k}\right) \cdot \frac{\left(\frac{k}{\mathrm{e}}\right)^k}{k^k} \geq \mathrm{e}^{-k}(1 - \mathrm{o}(1)) \ .
\end{aligned}$$

We now focus on phase 2 and condition on the events E1 through E5.

E6: The event that during the next n/k rounds no accepting mutation occurs and that at least one crossover choosing the correct two individuals is performed.

We first consider the mutations that could be harmful. Note that for such a mutation it is necessary to flip at least one 0-bit. Thus we calculate the probability that during each mutation none of the k 0s of any individual get flipped. The probability of this happening is at least $(1 - k/n)^{n/k} \geq \mathrm{e}^{-1}\big(1 - \mathrm{o}(1)\big)$.

We now look at the crossover. The probability of at least one crossover choosing the two correct individuals is at least $\big(1 - (1 - p_\mathrm{c})^{n/k}\big) \cdot \big(1/\binom{\mu}{2}\big) \geq (1 - \mathrm{e}^{-1})/\mu^2 \geq \mathrm{e}^{-1}\mu^{-2}$. The probability that the crossover creates the optimum is $2^{-2k}(1 - 1/n)^n \geq 2^{-2k}\mathrm{e}^{-1}$. All in all, the probability of E6 is at least $\mathrm{e}^{-3} \cdot 2^{-2k}\mu^{-2}\big(1 - \mathrm{o}(1)\big)$.

The probability of all of the events E1 through E6 happening is thus at least $\Omega(\mathrm{e}^{-5k}\mu^{-k-2})$, and the length of such an event is $n + n/k = \mathrm{O}(n)$. Hence, the expected time to create the optimum, once the plateau is reached, is in $\mathrm{O}(n\mathrm{e}^{5k}\mu^{k+2})$. □

4. DIVERSITY WITH DISTANCE

If we have $\mu \leq n/k$, then it is possible for the population to be perfectly spread, i.e., any two individuals do not share a position with a 0. We will see that this state is obtained quickly with three different diversity mechanisms.

Once any two individuals do not share a position with a 0, every crossover operation has a probability of 4^{-k} to generate the optimum.

4.1 Convex Hull Maximisation

Given two bit strings $x, y \in \{0,1\}^n$, uniform crossover can produce any bit string z such that, for all $i \leq n$, $z_i \in \{x_i, y_i\}$; in this sense, any such z is in between x and y. Accordingly one can define the convex hull of a set $P \subseteq \{0,1\}^n$ as the set of all those bit strings which are producible with repeated application of uniform crossover. In this sense, evolutionary search with crossover means searching the convex hull of the population [13].

Thus, it makes sense to consider a tie-breaking rule which maximizes the size of this convex hull. Since the size of the convex hull of a set P of bit strings is determined by the number of positions i for which there is an $x \in P$ with $x_i = 0$ and a $y \in P$ with $y_i = 1$, we can formalize this tie-breaking rule as follows. Given a population P of bit strings with worst fitness, remove an individual $z \in P$ such that

$$\sum_{i=1}^{n} [\exists x, y \in P \setminus \{z\} \colon x_i = 0 \wedge y_i = 1]$$

is maximised, where $[B]$ denotes the Iverson bracket (indicator function) for a proposition B.

Theorem 4.1. *Consider the (μ+1) GA with $\mu \leq n/k$ and $p_\mathrm{c} = 1 - \Omega(1) > 0$. Suppose that ties in the selection procedure are handled by maximising the convex hull. Then the expected number of iterations until an optimal individual is created when running on Jump_k, $k \leq n/2$, is $\mathrm{O}(\mu n^2 \log n + 4^k p_\mathrm{c}^{-1})$.*

Proof. First we determine the time needed until all individuals are on the plateau. By Lemma 2.1, this is $\mathrm{O}(\mu n + n \log n + \mu \log \mu)$. Note that $\mu \log \mu$ gets dominated by μn because we bound μ by a polynomial in n.

The rest of this proof is similar to the one of Theorem 3 in [9]. We first introduce some terms that come in handy for the rest of the proof.

We define *bad* 0s as 0s at a bit position such that there exists another individual having a 0 at that same position. We do not want to have such 0s, because they contradict our goal of all individuals ultimately having their 0s at unique positions.

Analogously we define a *good* 1 as a 1 at a bit position such that there exists no individual that has a 0 at that same position. We call such a position *good* as well. During mutation, good 1s can be used to turn into 0s that no other individual has.

We proceed via drift analysis and define our potential h to be the number of good positions. Let h' denote the potential after one iteration of the algorithm, and let A denote the event that $h' < h$. Note that h' cannot increase as each good position decreases the value of the convex hull by one, thus decreasing the convex hull if we end up with more good positions than before. So we can easily estimate

$$\mathrm{E}[h - h' \mid h] \geq \mathrm{E}[h - h' \mid h, A] \Pr(A \mid h) \geq \Pr(A \mid h) \ .$$

We decompose the analysis of $\Pr(A \mid h)$ into four smaller events.

E1: The event to choose an individual with a bad 0 for mutation. If total diversity has not been reached, there is at least one individual having a bad 0. So the probability of E1 is at least $1/\mu \cdot (1 - p_\mathrm{c})$.

E2: The event that the mutation flips exactly one 0 and exactly one 1 (thus the mutated individual is on the

plateau). Note that E2 does not focus on the 0 being a bad one and the 1 being a good one. The probability of any two-bit-flip mutation is $k/n \cdot (n-k)/n \cdot (1-1/n)^{n-2}$.

E3: The event, conditional on E1 and E2, to choose a bad 0 for mutation. The probability to do so is at least $1/k$.

E4: The event, conditional on E1 and E2, to choose a good 1. Due to the definition of h, the probability of E4 is $h/(n-k)$.

Taking all of this together, we get that the probability of A, given h, is at least

$$\frac{1-p_c}{\mu} \cdot \frac{1}{n} \cdot \frac{h}{n} \cdot \left(1-\frac{1}{n}\right)^{n-2} = \Omega\left(\frac{h}{\mu n^2}\right) .$$

Using the multiplicative drift theorem [7], we get a run time of $O(\mu n^2 \log n)$ until reaching maximal diversity.

At last, time needed to perform crossover after reaching maximum diversity takes expected p_c^{-1} rounds. Since all two different individuals have no 0s in common, such a crossover is successful with probability $1/2^{2k}$. This results in an overall waiting time of $4^k p_c^{-1}$ in the end. □

4.2 Total Hamming Distance

In the section before we looked at the maximisation of the convex hull of a population P. The convex hull operator only looks, per position, for two individuals having different bits at said position.

A more thorough operator can take all the bits per individual into account to give a more detailed view on the diversity of P. Such an operator is the one maximising the total Hamming distance of all individuals in P.

For any set of bit strings P we let

$$g(P) = \sum_{x \in P} \sum_{y \in P} d_H(x,y)$$

be the total Hamming distance of P. We consider the tie-breaking rule which, given a population P, removes an individual $z \in P$ such that $g(P \setminus \{z\})$ is maximised.

Theorem 4.2. *Consider the (μ+1) GA with $\mu < n/(2k)$ with $p_c = 1-\Omega(1) > 0$. Suppose that ties in the selection procedure are handled by maximising total Hamming distance. Then the expected number of iterations until an optimal individual is created when running on Jump_k, $k \le n/8$, is $O(n \log n + \mu^2 kn \log(\mu k) + 4^k p_c^{-1})$.*

Proof. This proof is similar to the one of Theorem 4.1 and uses the same terms of bad 0s and good 1s.

Using Lemma 2.1, all individuals are on the plateau within $O(\mu n + n \log n + \mu \log \mu)$ steps. The $\mu \log \mu$ term is dominated by μn because of our bound on μ.

The remaining analysis is, again, done via a drift argument. Note that the maximum of $g(P)$ is $2k\mu(\mu-1)$; thus, we let our potential function h be $2k\mu(\mu-1) - g(P)$ and show that this potential reaches 0.

The initial potential is at most $2k\mu(\mu-1)$ and h cannot increase, due to the selection operator always choosing an individual to discard such that the total Hamming distance is not decreased.

As before, let h' denote the potential after a mutation and let A denote the event that the potential decreased. We have $E[h - h' \mid h] \ge \Pr(A \mid h)$. Again, we decompose $\Pr(A \mid h)$.

E1: The event that an individual having a bad 0 is chosen for mutation. Each bad 0 adds at most $2(\mu-1)$ to h. Since each individual can have up to k bad 0s, there are at least $h/\big(2k(\mu-1)\big)$ individuals having at least one bad 0. The probability of E1 is therefore at least $h/\big(2k\mu(\mu-1)\big) \cdot (1-p_c)$.

E2: The event that mutation creates an individual on the plateau by flipping a 0- and a 1-bit. The probability of E2 is thus at least $k/n \cdot (n-k)/n \cdot (1-1/n)^{n-2}$.

E3: The event, conditioned on E1 and E2, to choose a bad 0 during mutation. The respective probability is at least $1/k$.

E4: The event, conditioned on E1 and E2, to choose a good 1 during mutation. We pessimistically assume that the total Hamming distance ($= 2k\mu(\mu-1) - h$) divided by $(\mu-1)$ is the number of 1s that are no longer good because each good 1 adds $(\mu-1)$ to the total Hamming distance. The mutation must choose one of the remaining good 1s. Thus, the probability of E4 is at least

$$\frac{n-k-\frac{2k\mu(\mu-1)-h}{\mu-1}}{n-k} = \frac{n+\frac{h}{\mu-1}-k\,(2\mu+1)}{n-k} .$$

Note that the numerator is always nonnegative for $\mu < n/(2k)$. Because we need $\mu \ge 3$, it follows that $k \le n/8$.

Overall, the probability of A is at least

$$\frac{h(1-p_c)}{2k\mu(\mu-1)} \cdot \frac{1}{n} \cdot \frac{n+\frac{h}{\mu-1}-k\,(2\mu+1)}{n} \cdot \left(1-\frac{1}{n}\right)^{n-2}$$
$$\ge \frac{h(1-p_c)\left(1+\frac{h}{(\mu-1)n}-\frac{2\mu+1}{n}k\right)}{2\mu^2 kn} e^{-1} \ge \frac{h(1-p_c)}{2\mu^2 kn} e^{-1} ,$$

which is positive as long as maximal diversity ($h = 0$) has not been reached. Note that this is our desired drift and that $1 - p_c = \Omega(1) > 0$.

We can now bound the expected time until maximal diversity has been reached by using the multiplicative drift theorem [7]. This yields an expected number of rounds in $O\big(\mu^2 kn \log(\mu k)\big)$.

Now all that is left is that a crossover is performed that generates the optimum. Again, the expected number of steps for this event is $4^k p_c^{-1}$. This completes the proof. □

4.3 Fitness Sharing

We consider a tie-breaking rule that makes use of fitness sharing as described in [8]. The actual fitness $f(x)$ of an individual x gets skewed by how similar it is with respect to a certain measure d to other individuals. The new shared fitness $\bar{f}(x)$ is the basis of the tie-breaking mechanism, where we just delete one individual with worst shared fitness.

The general scheme of the fitness sharing mechanism is parameterised with a metric d and two numbers α and σ. Given a population P of individuals with worst fitness, we want to remove an individual $z \in P$ such that

$$\sum_{x \in P \setminus \{z\}} \frac{f(x)}{\sum_{y \in P \setminus \{z\}} \max\left\{0, 1-\left(\frac{d(x,y)}{\sigma}\right)^{\alpha}\right\}}$$

is maximised. That means that there is a penalty for the similarity of x to all other individuals y up to a distance of σ; α determines the shape of the penalty.

In this paper we consider the Hamming distance, i.e., $d = d_H$, and we set $\alpha = 1$, that is, we have a linear penalty. Note that we only use this fitness sharing rule for breaking ties (this ensures that the initial climb to the plateau is undisturbed). We call this tie-breaking rule *Hamming fitness sharing*.

Lemma 4.3. *Given a population P of individuals all having the same fitness; suppose that any two individuals of P differ by at most σ. Then the Hamming fitness sharing tie-breaking rule maximises the total Hamming distance of P.*

Proof. Since all individuals $x \in P$ have the same fitness, we can ignore the impact of the fitness on the tie-breaking rule. Furthermore, since individuals can differ by at most σ, the maximum in the fitness sharing expression is not necessary. So we want to remove an individual $z \in P$ such that $\sum_{x \in P \setminus \{z\}} \left(\sum_{y \in P \setminus \{z\}} (1 - \frac{d_{\mathrm{H}}(x,y)}{\sigma}) \right)^{-1}$ is maximised. That is the same as *minimising* the term $\sum_{x \in P \setminus \{z\}} \sum_{y \in P \setminus \{z\}} -d_{\mathrm{H}}(x,y)$ since we just change the monotony and remove the offset of $\sum_{y \in P \setminus \{z\}} 1$ and the common factor $1/\sigma$; this is just the same as maximising the total Hamming distance of P. □

Thanks to this lemma, we now know that Hamming fitness sharing and maximising the Hamming distance are equivalent. Thus, we can immediately carry over all run time results from the maximisation of the Hamming distance to fitness sharing, as stated in the following theorem.

Theorem 4.4. *Consider the (μ+1) GA with $\mu < n/(2k)$ with $p_{\mathrm{c}} = 1 - \Omega(1) > 0$. Suppose that ties in the selection procedure are handled by Hamming fitness sharing with $\sigma \geq 2k$. Then the expected number of iterations until an optimal individual is created when running on Jump_k, $k \leq n/8$, is $\mathrm{O}(n \log n + \mu^2 kn \log(\mu k) + 4^k p_{\mathrm{c}}^{-1})$.*

Proof. The time needed to reach the plateau is by Lemma 2.1 in $\mathrm{O}(\mu n + n \log n + \mu \log \mu)$. As we bound μ by a polynomial in n, the $\mu \log \mu$ term is dominated by μn.

After reaching the plateau, Lemma 4.3 yields that on the plateau Hamming fitness sharing is just maximisation of total Hamming distance. Thus, the statement follows from the proof of Theorem 4.2. □

5. ISLAND MODEL

An easy way to ensure diversity in a population is to keep different parts strictly separate. Previous results [14] have shown that island models can provide enough diversity that can be subsequently leveraged by crossover. We consider a single-receiver island model, described in [16], as follows. We have $\mu + 1$ islands, μ of them running a (1+1) EA independently from one another. Furthermore, there is one island which is the receiver island. Each iteration it chooses two of the μ islands uniformly at random, copies their respective best-so-far individuals, and then performs uniform crossover on those, hence $p_{\mathrm{c}} = 1$. The resulting offspring replaces the resident individual if it has higher fitness. We say that the island model succeeds if the receiver island produces the optimum. We do not employ any particular tie-breaking rule in the islands (and, in the case of a tie, choose a survivor uniformly at random).

Theorem 5.1. *Consider the island model with $p_{\mathrm{c}} = 1$, $\mu = \mathrm{O}(n^c)$ islands for any c, and $\mu \geq 2$. For optimizing Jump_k, $k = \mathrm{o}(\sqrt{n}/\mu)$, the expected run time until the optimum is produced is $\mathrm{O}\big(n \log n + \mu^2 kn + \mu^2 4^k\big)$.*

Proof. In this proof, we follow the same language as in the proof of Theorem 4.1.

In expectation, after $\mathrm{O}\big(n \log n^d\big)$ steps, for a constant $d > 0$, a single (1+1) EA is on the plateau with probability $1 - 1/n^d$ [6]. So the probability of μ independently running (1+1) EAs being *all* on the plateau after $\mathrm{O}\big(n \log n^d\big)$ steps is $\big(1 - 1/n^d\big)^\mu$, which goes toward 1 as n goes to infinity from of our constraint on μ and because we can choose $d > c$.

Once all (1+1) EAs are on the plateau, we proceed via drift analysis: we define a potential for the island model and show that there is a bias toward 0. Fix two islands under consideration. The potential X_t at point t (starting from the plateau) is defined as follows: it is 0 if the receiver island produced the optimum, else we have a look at the individuals of the two fixed islands. Assume that these individuals have $i \leq k$ of their 0s in common. X_t is then $i\mu^2 n\mathrm{e}^2 + \mu^2 4^k \mathrm{e}$.

The drift is the expected change in potential, i.e., $\mathrm{E}(X_t - X_{t+1} \mid X_t)$. We make a case distinction to whether the potential decreases or increases. In order to compute the drift, we assume $X_t > 0$ to be given. Let i be such that $X_t = i\mu^2 n\mathrm{e}^2 + \mu^2 4^k \mathrm{e}$.

First we consider $i > 0$. Whenever the potential decreases, it decreases by at least $\mu^2 n\mathrm{e}^2$.

We now lower-bound the probability of the potential decreasing, i.e., the two individuals increase their Hamming distance: we assume that only one individual flips a bad 0 and a good 1 and that our two individuals are chosen. No other bits are mutated. Therefore, the probability is at least

$$\frac{2}{\mu^2} \left(1 - \frac{1}{n}\right)^{n-2} \cdot \frac{n - 2(k-i) - i}{n} \cdot \frac{i}{n} \left(1 - \frac{1}{n}\right)^{n} \geq \frac{1}{\mu^2 n\mathrm{e}^2} \ .$$

Thus, the overall positive drift is at least 1.

If we now consider $i = 0$, we easily get a positive drift of at least 1 as well, since then all that is left to create the optimum is to have the uniform crossover always choose the correct bits of the $2k$ positions where the two individuals differ, and not mutating any bit after crossover, resulting in a probability of at least $1/(\mu^2 4^k \mathrm{e})$.

We now upper-bound the negative drift. We only consider the case that i increases by 1. An increase by more than 1 is possible but far more unlikely since more bit flips have to occur, leading to an additional factor of $1/n^2$ in the probability for each additional 1. We make up for the so lost terms by multiplying our negative drift with a constant c.

The change in potential conditioned on a decrease as we defined it is $-\mu^2 n\mathrm{e}^2$, and the probability of such a decrease is at most ck^2/n^2: each of the two individuals can only decrease the distance to the other one by flipping one of its own 0s and flipping a 1 where the other individual has a 0. The absolute value of the negative drift is thus at most $c(\mu k\mathrm{e})^2/n$. Using our assumption regarding k, this results in a negative drift in o(1). The additive drift theorem [10] hence yields an expected run time of $\mathrm{O}\big(\mu^2 kn + \mu^2 4^k\big)$ when starting from the plateau. Taking the time needed for all islands to be on the plateau into account, we get the desired run time of $\mathrm{O}\big(n \log n + \mu^2 kn + \mu^2 4^k\big)$. □

6. EXPERIMENTS

Since the theoretical results presented in the previous section are asymptotic and they mostly provide upper bounds on the run time of the algorithms, we also implemented the (μ+1) GA and compared the theoretically analysed diversity mechanisms experimentally. The population size is set to $\mu = 4\mathrm{e} \ln n$. Full crossover is enabled ($p_{\mathrm{c}} = 1.0$) and the

problem size n is varied in $[100, 1000]$ (with a step size of 25). In each tested setting, the run is replicated 100 times with different random seeds and the number of function evaluations, denoted as '# evaluations', is reported as the run time. The result for $k = 4$ is shown in Figure 1. Note that we also did the experiment with $p_c = 0$ for the no-mechanism setting, e.g., comparing the EA with the GAs, however, the average run time for $n = 100$ in this experiment is already $2.28 \cdot 10^8$, which cannot be displayed in the figure.

On average, the highest contribution to the reduction of the run time in order is fitness sharing, then convex hull maximisation, deterministic crowding, and, finally, duplicate elimination and minimisation have quite similar average run times. We also notice that the the island model with $\mu = 2$ requires approximately the same average numbers of evaluations as deterministic crowding. Overall, compared to the standard (μ+1) GA, all the diversity mechanisms contribute to the reduction of the average run time, as well as to the stability of the result.

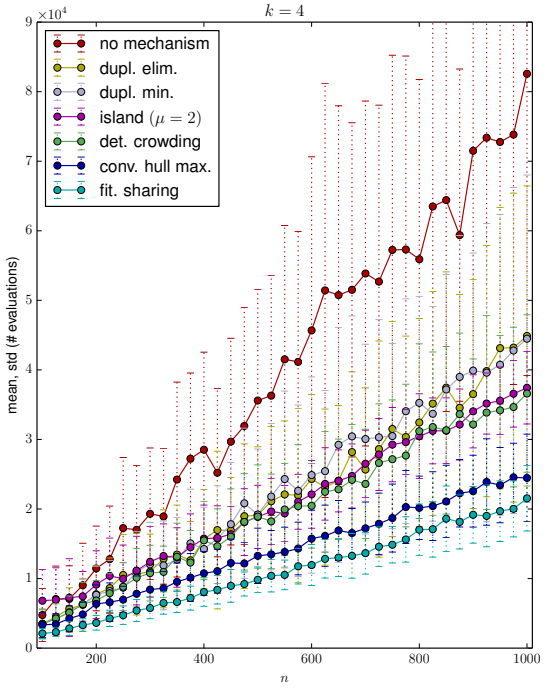

Figure 1: Performance of the diversity mechanisms.

7. CONCLUSION

We have considered the role of selection-based diversity mechanisms used together with crossover for escaping local optima. We prove rigorous upper bounds on the run time of the (μ+1) GA for seven well-known diversity mechanisms optimising the Jump_k function. Our results reveal a qualitative difference in the ability of the different diversity mechanisms to escape local optima.

In contrast to previous theoretical work on crossover for Jump_k, our upper bounds do not rely on unreasonably small (e.g., vanishing with n) crossover probabilities, but instead cover the more practical case of constant crossover probabilities. Furthermore, our proofs provide insight into the ways that diversity mechanisms, when applied as a tie-breaking rule in selection, can quickly spread the population out over the jump plateau in order to get enough diversity for crossover to combine the correct solution components to escape the set of local optima.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 618091 (SAGE) and from the EPSRC under grant no. EP/M004252/1.

References

[1] E. Alfaro-Cid, J. J. M. Guervós, F. F. de Vega, A. I. Esparcia-Alcázar, and K. Sharman. Bloat control operators and diversity in genetic programming: A comparative study. *Evol. Comput. Journal*, 18:305–332, 2010.

[2] J. Arabas. Approximating the genetic diversity of populations in the quasi-equilibrium state. *IEEE Trans. Evol. Comput.*, 16:632–644, 2012.

[3] G. Bell. *The masterpiece of nature the evolution and genetics of sexuality*. 1982.

[4] E. K. Burke, S. M. Gustafson, and G. Kendall. Diversity in genetic programming: an analysis of measures and correlation with fitness. *IEEE Trans. Evol. Comput.*, 8:47–62, 2004.

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

[6] B. Doerr and L. A. Goldberg. Adaptive drift analysis. *Algorithmica*, 65:224–250, 2013.

[7] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 64:673–697, 2012.

[8] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt. Analysis of diversity-preserving mechanisms for global exploration. *Evol. Comput. Journal*, 17:455–476, 2009.

[9] W. Gao and F. Neumann. Runtime analysis for maximizing population diversity in single-objective optimization. In *Proc. of GECCO '14*, pp. 777–784, 2014.

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

[11] T. Jansen and I. Wegener. The Analysis of Evolutionary Algorithms - a Proof That Crossover really can help. *Algorithmica*, 34:47–66, 2002.

[12] T. Kötzing, D. Sudholt, and M. Theile. How crossover helps in pseudo-boolean optimization. In *Proc. of GECCO '11*, pp. 989–996, 2011.

[13] A. Moraglio and D. Sudholt. Runtime analysis of convex evolutionary search. In *Proc. of GECCO '12*, pp. 649–656, 2012.

[14] F. Neumann, P. S. Oliveto, G. Rudolph, and D. Sudholt. On the effectiveness of crossover for migration in parallel evolutionary algorithms. In *Proc. of GECCO '11*, pp. 1587–1594, 2011.

[15] R. K. Ursem. Diversity-guided evolutionary algorithms. In *Proc. of PPSN VII*, pp. 462–474, 2002.

[16] R. A. Watson and T. Jansen. A building-block royal road where crossover is provably essential. In *Proc. of GECCO '07*, pp. 1452–1459, 2007.

[17] C. Witt. Runtime Analysis of the (μ+1) EA on Simple Pseudo-Boolean Functions. *Evol. Comput. Journal*, 14: 65–86, 2006.