This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

Escaping Local Optima Using Crossover with Emergent Diversity

Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Martin S. Krejca, Per Kristian Lehre, Pietro S. Oliveto, Dirk Sudholt, Andrew M. Sutton

***Abstract*—Population diversity is essential for avoiding premature convergence in Genetic Algorithms and for the effective use of crossover. Yet the dynamics of how diversity emerges in populations are not well understood. We use rigorous run time analysis to gain insight into population dynamics and Genetic Algorithm performance for the (μ+1) Genetic Algorithm and the** Jump **test function. We show that the interplay of crossover followed by mutation may serve as a catalyst leading to a sudden burst of diversity. This leads to significant improvements of the expected optimisation time compared to mutation-only algorithms like the (1+1) Evolutionary Algorithm. Moreover, increasing the mutation rate by an arbitrarily small constant factor can facilitate the generation of diversity, leading to even larger speedups. Experiments were conducted to complement our theoretical findings and further highlight the benefits of crossover on the function class.**

***Index Terms*—Genetic algorithms, recombination, diversity, run time analysis, theory**

I. Introduction

Genetic Algorithms (GAs) are powerful general-purpose optimisers that perform surprisingly well in many applications, including those where the problem is not well understood to apply a tailored algorithm. Their wide-spread success is based on a number of factors: using populations to diversify search, using mutation to generate novel solutions, and using crossover to combine features of good solutions.

Prügel-Bennett [29] gives several reasons for the success of populations and crossover. Crossover can combine building blocks of good solutions and help to focus the search on bits where parents disagree [29]. For both tasks, the population needs to be diverse enough; without sufficient diversity in the population, crossover is not effective. A common problem in the application of GAs is the loss of diversity when the population converges to copies of the same search point, often called *premature convergence*. Understanding how populations gain and lose diversity during the course of the optimisation is vital for understanding the working principles of GAs and for tuning the design of GAs to get the best possible performance.

Rigorous run time analysis has emerged as a powerful theory that has provided many insights into the performance of GAs [1], [5], [17], [24], [26], [27], including the benefit of crossover [9], [18], [20], [21], [25], [31]. It has guided algorithm design, including the discovery of new variants of GAs such as the (1+(λ,λ)) GA [8], which has shown very good performance across a range of hard problems [14].

However, understanding population diversity and crossover has proved elusive. The first example function where crossover was proven to be beneficial is called Jump_k. In this problem, GAs have to overcome a fitness valley such that all local optima have Hamming distance k to the global optimum. Jansen and Wegener [18] showed that, while mutation-only algorithms such as the (1+1) EA require expected time $\Theta(n^k)$, a simple (μ+1) GA with crossover only needs time $\mathrm{O}\big(\mu n^2 k^3 + 4^k/p_{\mathrm{c}}\big)$. This time is $\mathrm{O}\big(4^k/p_{\mathrm{c}}\big)$ for large k, and hence significantly faster than mutation-only GAs. However, their analysis requires an unrealistically small crossover probability $p_{\mathrm{c}} \le 1/(ckn)$ for a large constant $c > 0$.

Kötzing, Sudholt, and Theile [20] later refined these results towards a crossover probability $p_{\mathrm{c}} \le k/n$, which is still unrealistically small. Both approaches focus on creating diversity through a sequence of lucky mutations, relying on crossover to create the optimum, once sufficient diversity has been created. Their arguments break down if crossover is applied frequently. Hence, these analyses do not reflect the typical behaviour in GA populations with constant crossover probabilities $p_{\mathrm{c}} = \Theta(1)$ as used in practice [22].

Lehre and Yao analysed the run time of the (μ+1) GA with deterministic crowding for arbitrary crossover rates $p_{\mathrm{c}} > 0$, showing exponential run time gaps between the case $p_{\mathrm{c}} = 0$ and $p_{\mathrm{c}} > 0$ [21]. The gain in performance in that analysis stems from the ability of a diverse population to optimise multiple, separated paths in parallel using a diversity-preservation mechanism. Similar results have been also shown for instances of the vertex cover problem by generating diversity, either through deterministic crowding [25] or through island models [23]. Recently in [7], we have shown that a small change to the tie-breaking rule of the (μ+1) GA to introduce many common principles of preserving diversity can lead to a sizeable advantage on the expected optimisation time of Jump_k function. The results hold for realistic crossover probabilities $p_{\mathrm{c}} = 1 - \Omega(1)$. In the present paper, we will consider a very different effect.

We provide a novel approach loosely inspired from population genetics: we show that diversity can also be created by crossover, followed by mutation. Note that the perspective of crossover creating diversity is common in population genetics [19], [33]. A frequent assumption is that crossover mixes all alleles in a population, leading to a situation called *linkage*

D.-C. Dang is with the ASAP Research Group at the School of Computer Science, University of Nottingham, United Kingdom.

P. K. Lehre is with the School of Computer Science, University of Birmingham, United Kingdom.

T. Friedrich, T. Kötzing, M. S. Krejca and A. M. Sutton are with the Chair of Algorithm Engineering, Hasso Plattner Institute, Potsdam, Germany.

P. S. Oliveto and D. Sudholt are with the Department of Computer Science, University of Sheffield, United Kingdom.

equilibrium, where the state of a population is described by the frequency of alleles [3].

For the maximum crossover probability $p_c = 1$, we show that on Jump_k diversity emerges naturally in a population: the interplay of crossover, followed by mutation, can serve as a catalyst for creating a diverse range of search points out of few different individuals. This naturally emerging diversity allows proving a speedup of order $n/\log n$ for $k \geq 3$ and standard mutation rate $p_m = 1/n$ compared to mutation-only algorithms such as the (1+1) EA. Increasing the mutation rate to $p_m = (1+\delta)/n$ for an arbitrarily small constant $\delta > 0$, leads to a speedup of order n. The detail can be seen in Table I.

Both operators are proven to be vital: mutation requires $\Theta(n^k)$ expected iterations to hit the optimum from a local optimum. Also using crossover on its own does not help much. As shown in [20, Theorem 8], using only crossover with $p_c = \Omega(1)$ but no mutation following crossover, diversity reduces quickly, leading to inefficient running times for small population sizes ($\mu = O(\log n)$).

All our analyses are based on observing the dynamic behaviour of the size of the largest *species*, referring to a collection of identical genotypes as species. A population contains no diversity when only one species is present. However, mutation can create further species, and then the combination of crossover and mutation is able to rapidly create further species in a highly stochastic process. This diversity can then be exploited to find the global optimum on Jump_k efficiently. A higher mutation rate facilitates the generation of new species and leads to better performance, with respect to rigorous upper run time bounds and empirical performance.

Using Jump_k as a case study, our analyses shed light on how diversity emerges in populations and how to facilitate the emergence of diversity by tuning the mutation rate. The general proof strategy we take is as follows. We characterise the size of the largest species as a stochastic process and calculate the transition probabilities of this process taking into account both mutation and crossover. We prove that the size of the largest species is described either by an almost-fair random walk (for standard mutation rates), or by an unfair random walk that is biased toward increased diversity (for higher mutation rates). This ultimately allows us to bound the expected time until sufficient diversity is present in the population to perform a crossover that successfully generates the global optimum. Our main results are stated in Theorems 6 and 10, which yield our run time bounds under the assumed conditions. Critical lemmas are Lemma 2, which estimates the time until the entire population has reached the plateau using the method of fitness-based partitions, and Lemma 4, which bounds the transition probabilities for the random walk dynamics of the size of the largest species. The proof of Lemma 4 is carried out by a careful analysis of the different events that can occur while the entire population resides on the plateau.

This work is based upon our preliminary study published in [6]. Here we extend the analysis to higher mutation rates, leading to the surprising conclusion that increasing the mutation rates leads to smaller run time bounds, compared to the standard mutation rate $1/n$. Furthermore, the analysis of standard mutation rates in [6] was restricted to very short jumps, $k = O(1)$. Here we generalise the results to a much larger class of Jump_k functions, only requiring $k = o(n)$. Experiments were conducted to complement the theoretical results and further highlight the benefits of combining crossover with mutation. In fact, the experimental results showed that the setting of high mutation rate can be as competitive as using specific diversity mechanisms from [7].

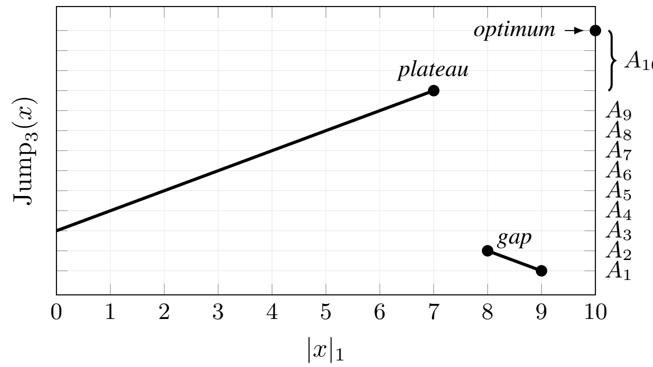

Figure 1. Illustration of the Jump_k fitness function for the case $n = 10$ and $k = 3$, including the levels $A_1, \ldots, A_{10}$ defined in the proof of Lemma 2.

II. Preliminaries

The $\text{Jump}_k \colon \{0,1\}^n \to \mathbb{N}$ class of pseudo-Boolean fitness functions was originally introduced by Jansen and Wegener [18]. The function value increases with the number of 1-bits in the bit string until a *plateau* of local optima is reached, consisting of all points with $n-k$ 1-bits. However, its only global optimum is the all-ones string 1^n. Between the plateau and the global optimum, there is a valley of bad fitness, which we call the *gap* of length k, and the algorithm has to jump over this gap to optimise the function.

The function is formally defined as

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

where $|x|_1 = \sum_{i=1}^n x_i$ is the number of 1-bits in x. Figure 1 illustrates the function, with the number of 1-bits on the horizontal axis, and the function value on the vertical axis.

We will analyse the performance of a standard steady-state (μ+1) GA [18] using uniform crossover (i.e., each bit of the offspring is chosen uniformly at random from one of the parents) and standard bit mutation (i.e., each bit is flipped with probability p_m). The algorithm uses a population of μ individuals. In each generation, a new individual is created. With probability p_c, it is created by selecting two parents from the population uniformly at random, crossing them over, and then applying mutation to the resulting offspring. With probability $1 - p_c$ instead, one single individual is selected and only mutation is applied. The generation is concluded by removing the worst individual from the population and breaking ties uniformly at random. Algorithm 1 shows the pseudocode for the (μ+1) GA. Note that P is a multiset.

The most interesting behaviour of the population presented in this paper occurs after the entire population is stuck at

Table I
SOME EXAMPLES OF RUN TIME BOUNDS WE OBTAIN FOR THE (μ+1) GA ON Jump_k.

Algorithmic setting	$\boldsymbol{\mu}$	$\boldsymbol{p_\mathrm{c}}$	$\boldsymbol{p_\mathrm{m}}$	**Problem**		
				$\boldsymbol{k=2}$	$\boldsymbol{k=4}$	$\boldsymbol{k=\mathrm{o}(n)}$
Standard mutation, Thm. 6	$\Theta\left(\frac{\sqrt{n}}{\sqrt{\log n}}\right)$	1	$1/n$	$\mathrm{O}\left(\frac{n^{1.5}}{\sqrt{\log n}}\right)$	$\mathrm{O}\left(\frac{n^{3.5}}{\sqrt{\log n}}\right)$	$\mathrm{O}\left(\frac{n^{k-0.5}}{\sqrt{\log n}}\right)$
	$\Theta(n)$	1	$1/n$	$\mathrm{O}(n^2 \log n)$	$\mathrm{O}(n^3 \log n)$	$\mathrm{O}(n^{k-1} \log n)$
High mutation, Thm. 10	$\Theta(\log n)$	1	$(1+\delta)/n$	$\mathrm{O}(n \log n \log\log n)$	$\mathrm{O}(n^3)$	$\mathrm{O}\left(n\sqrt{k} \log n \log\log n + n^{k-1}\right)$

Algorithm 1: (μ+1) GA

```
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, p_m);
10     P ← P ∪ {z};
11     Remove one element from P with lowest fitness,
         breaking ties uniformly at random;
```

local optima, the so-called plateau. That is because under the right condition the population diversity will emerge during this stage. Then after sufficient progress is made in diversity, crossover and mutation can work together on the plateau to create an optimal solution in $\mathrm{o}(n^k)$ time. This is captured by Lemma 14, which will be presented later in the paper.

For the sake of completeness, in the next section, we provide the time bounds for the population to reach the plateau for the general setting of $p_\mathrm{c} = \Omega(1)$. This covers the case of $p_\mathrm{c} = 1$ which we will actually focus on in the main results.

III. TIME TO PLATEAU

In the setting of $p_\mathrm{c} = \Omega(1)$, we direct the attention to the steps that crossover occurs. We make use of the following general result, which provides an upper bound on the expected time for the (μ+1) GA to reach some region A_m of the search space. Here we consider a *fitness-based* partition (see [17] for a formal definition) into levels $(A_i)_{i\in[m]}$ (thus, A_m is the last level) and define $A_{\geq j} := \bigcup_{i=j}^{m} A_i$.

Theorem 1. *Let $(A_i)_{i\in[m]}$ be a fitness-based partition of the search space into $m \in \mathbb{N}$ levels. If there exist parameters $\varepsilon, s_1, \ldots, s_{m-1} \in (0,1]$ such that for all $j \in [m-1]$*

1) $\min_{x\in A_{\geq j}, y \in A_{\geq j+1}}$ $\Pr(\mathrm{mutate}(\mathrm{crossover}(x,y)) \in A_{\geq j+1}) \geq \varepsilon$ *and*
2) $\min_{x,y\in A_j} \Pr(\mathrm{mutate}(\mathrm{crossover}(x,y)) \in A_{\geq j+1}) \geq s_j$

then the expected number of iterations until the entire population of the (μ+1) GA with $p_\mathrm{c} = \Omega(1)$ is in A_m is $\mathrm{O}\left((\mu m/\varepsilon)\log(\mu) + \sum_{j=1}^{m-1} 1/s_j\right)$.

Proof. The proof follows [5], but we avoid a detailed drift analysis because the algorithm is elitist, i.e., the maximum fitness in the population does not decrease. Let the *current level* be the smallest $i \in [m]$ such that the population contains less than $\mu/2$ individuals in $A_{\geq i+1}$. By definition, there are at least $\mu/2$ individuals in $A_{\geq j}$, where j is the current level.

Since the algorithm is elitist, the number of individuals in $A_{\geq j}$ is non-decreasing for any $j \in [m]$. For an upper bound, we ignore any improvements where mutation only is used (i.e., lines 8 and 9 in Alg. 1).

Assume that there are i individuals in $A_{\geq j+1}$, hence $0 \leq i < \mu/2$. If $i = 0$, then an individual in $A_{\geq j+1}$ can be created by selecting two individuals from A_j, crossing them over, and mutating them such that the offspring is in $A_{\geq j+1}$ and an individual not in $A_{\geq j+1}$ is removed. The probability of this event is at least $p_\mathrm{c} s_j/4$, where the $1/4$ is the probability of selecting two individuals from A_j, which contains at least $\mu/2$ individuals.

If $0 < i < \mu/2$, then the number of individuals in $A_{\geq j+1}$ can be increased by selecting an individual in $A_{\geq j}$ and an individual in $A_{\geq j+1}$, crossing them over, and mutating them such that the offspring is in $A_{\geq j+1}$ and one of the $\mu - i > \mu/2$ individuals not in $A_{\geq j+1}$ is removed. This event occurs with probability at least $(p_\mathrm{c}/2)(i/\mu)\varepsilon$.

The expected time to increase the number of individuals in $A_{\geq j+1}$ from 0 to $\mu/2$, i.e., to increase the current level by at least one, is $4/(p_\mathrm{c} s_j) + 2\mu/(p_\mathrm{c}\varepsilon)\sum_{i=1}^{\mu/2} 1/i$. Hence, the expected time until at least half of the population is in A_m is $\mathrm{O}\left((\mu m/\varepsilon)\log(\mu) + \sum_{j=1}^{m-1} 1/s_j\right)$.

We now consider the time to remove individuals from the lowest fitness level in the population, assuming that at least half of the population has reached the last level A_m. Assume that there are $0 < i' < \mu/2$ individuals in the lowest level $j < m$. The number of individuals in level j can be reduced by crossing over an individual in level j and one of the at least $\mu/2$ individuals in level m, and mutating the offspring so that it belongs to $A_{\geq j+1}$. By condition 1, this event occurs with probability at least $p_\mathrm{c}(\varepsilon/2)(i'/\mu)$. Hence, the expected time to remove all individuals from the lowest level j is no more than $(2/p_\mathrm{c}\varepsilon)\mu\sum_{i'=1}^{\mu/2} 1/i' = \mathrm{O}((\mu/\varepsilon)\log\mu)$. The expected time until all individuals in fitness levels lower than m have been removed is therefore $\mathrm{O}(\mu(m/\varepsilon)\log\mu)$. □

We apply Theorem 1 to bound the time until the entire population reaches the plateau.

Lemma 2. *Consider the (μ+1) GA optimising* Jump_k *with* $p_\mathrm{c} = \Omega(1)$ *and* $p_\mathrm{m} = \Theta(1/n)$. *Then the expected time until either the optimum has been found or the entire population is on the plateau is* $\mathrm{O}\left(n\sqrt{k}(\mu \log \mu + \log n)\right)$.

Proof. We divide the search space into $m := n$ fitness levels with the partition

$$A_j := \begin{cases} \{x \in \{0,1\}^n \mid |x|_1 = n - j\} & \text{if } 1 \le j < k, \\ \{x \in \{0,1\}^n \mid |x|_1 = j - k\} & \text{if } k \le j < n, \\ \{x \in \{0,1\}^n \mid |x|_1 \in \{n-k, n\}\} & \text{if } j = n. \end{cases}$$

We call any search point $x \in \{0,1\}^n$ with $n-k < |x|_1 < n$ a *gap-point*. Gap-points have worse fitness than any other search point, hence once there are no gap-points left in the population, the algorithm will not accept any further gap-points. We can therefore divide the run into two phases, with phase 1 lasting as long as the population contains at least one gap-individual, followed by phase 2, which lasts until the optimum or a plateau individual has been found.

We bound the duration of the two phases by applying Theorem 1 twice, once for each of the two phases.

We start by estimating the expected duration of phase 2 using Theorem 1 with respect to levels A_k to level A_n. We claim that the probability of producing a gap-point by crossing over two individuals $x \in A_{\ge j}$ and $y \in A_{\ge j+1}$ with $k \le j < n+1$ satisfies

$$\Pr(n - k < |\mathrm{crossover}(x,y)|_1 < n) < \frac{1}{2} - \frac{1}{4\sqrt{k}} \ . \tag{1}$$

To see why this claim holds, we first argue that the probability of producing a gap-point is highest when both parents, x and y, have $n-k$ 1-bits. More formally, obtain x' by flipping an arbitrary 0-bit in x, and y' by flipping an arbitrary 0-bit in y. Then, we have the stochastic dominance relationships $|\mathrm{crossover}(x,y)|_1 \preceq |\mathrm{crossover}(x',y)|_1$ and $|\mathrm{crossover}(x,y)|_1 \preceq |\mathrm{crossover}(x,y')|_1$. By repeating this argument, we obtain $|\mathrm{crossover}(x,y)|_1 \preceq |\mathrm{crossover}(x'',y'')|_1$ for two bit strings x'' and y'' with $|x''|_1 = |y''|_1 = n-k$. The probability of obtaining a search point with exactly k 0-bits when crossing over two bit strings with k 0-bits each is minimised when all positions of the 0-bits in the two bit strings differ. Hence, for bit strings x'' and y'', we have by Stirling's approximation the lower bound

$$\begin{aligned} \Pr(|\mathrm{crossover}(x'',y'')|_1 = n-k) &> \binom{2k}{k} \cdot 2^{-2k} \\ &\ge \frac{2^{2k}}{2\sqrt{k}} \cdot 2^{-2k} = \frac{1}{2\sqrt{k}} \ . \end{aligned}$$

Uniform crossover of the bit strings x'' and y'' creates two bit strings u'' and v'', and returns either u'' or v'' with equal probability. We then have

$$2(n-k) = |x''|_1 + |y''|_1 = |u''|_1 + |v''|_1 \ . \tag{2}$$

The event $|u''|_1 = |v''|_1 = n-k$ therefore equals the event $|\mathrm{crossover}(x'',y'')|_1 = n-k$. Otherwise, in the event that $|u''|_1 \ne |v''|_1$, we assume without loss of generality that $|u''|_1 > |v''|_1$. We must then have $|v''|_1 < n-k < |u''|_1$ because

$$2|v''|_1 < |u''|_1 + |v''|_1 = 2(n-k) = |u''|_1 + |v''|_1 < 2|u''|_1.$$

The claimed inequality (1) now follows, because

$$\begin{aligned} &\Pr(n > |\mathrm{crossover}(x'',y'')|_1 > n-k) \\ &\qquad < \Pr(|\mathrm{crossover}(x'',y'')|_1 > n-k) \\ &\qquad = \frac{1}{2}\Pr(|u''|_1 \ne |v''|_1) \\ &\qquad = \frac{1}{2}(1 - \Pr(|u''|_1 = |v''|_1)) \\ &\qquad = \frac{1}{2}(1 - \Pr(|\mathrm{crossover}(x'',y'')|_1 = n-k)) \\ &\qquad \le \frac{1}{2} - \frac{1}{4\sqrt{k}} \ . \end{aligned}$$

We now show that condition 1 of Theorem 1 holds for the parameter $\varepsilon := (1-p_\mathrm{m})^n/(4\sqrt{k}) = \Theta(1/\sqrt{k})$. Assume that $x \in A_{\ge k+j}$ and $y \in A_{\ge k+j+1}$ for $j \ge 0$. By the same arguments as above,

$$2j + 1 \le |x|_1 + |y|_1 = |u|_1 + |v|_1 \le 2|u|_1 \ ,$$

where we assume without loss of generality that $|u|_1 \ge |v|_1$. A crossover between x and y therefore produces two offspring u and v where $|u|_1 \ge j+1$, hence

$$\Pr(j + 1 \le |\mathrm{crossover}(x,y)|_1) \ge 1/2 \ . \tag{3}$$

Combining (1) and (3) now yields

$$\begin{aligned} &\Pr(\mathrm{crossover}(x,y) \in A_{\ge k+j+1}) \\ &\qquad = \Pr(j+1 \le |\mathrm{crossover}(x,y)|_1) \\ &\qquad\quad - \Pr(n-k < |\mathrm{crossover}(x,y)|_1 < n) \\ &\qquad \ge \frac{1}{4\sqrt{k}} \ . \end{aligned}$$

Finally, with probability $(1-p_\mathrm{m})^n$, none of the bits are flipped during mutation, which implies

$$\Pr(\mathrm{mutate}(\mathrm{crossover}(x,y)) \in A_{\ge k+j+1}) \ge \varepsilon \ .$$

We now show that condition 2 of Theorem 1 holds. Assume that $x, y \in A_{j+k}$ for $j \ge 0$. Then, following the same arguments as above

$$\Pr(\mathrm{crossover}(x,y) \in A_{\ge k+j}) \ge \frac{1}{4\sqrt{k}} \ .$$

The probability that the mutation operator flips at least one of the $n-j$ 0-bits, and no other bits, is at least $(n-j)p_\mathrm{m}(1-p_\mathrm{m})^{n-1}$. Hence, we can use the parameter $s_j := (n-j)p_\mathrm{m}(1-p_\mathrm{m})^{n-1}/(4\sqrt{k}) = \Theta((n-j)/(n\sqrt{k}))$.

We have shown that both condition 1 and condition 2 hold during phase 2, which by Theorem 1 implies that the expected duration of phase 2 is $\mathrm{O}\left(n\sqrt{k}(\mu \log \mu + \log n)\right)$.

To estimate the expected duration of phase 1, we again apply Theorem 1, but this time with respect to level A_1 to level A_k. We can reuse the bounds from phase 2, except that we count the number of 0-bits rather than the number of 1-bits, and we do not need to account for the probability of producing gap individuals. Hence, we obtain the same upper bound for the

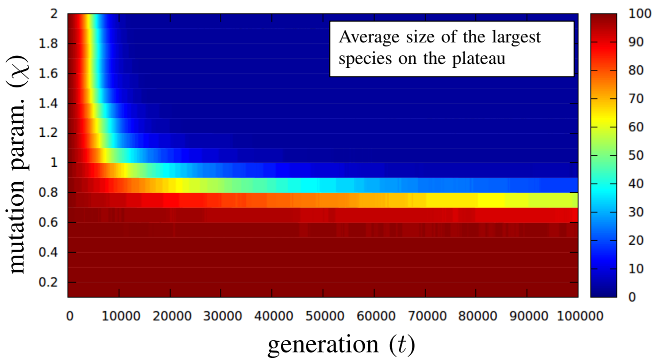

Figure 2. Empirical investigation of diversity on the plateau.

expectation duration of phase 1 and phase 2, and the theorem follows. □

In the following sections, we first show that once the plateau of Jump_k has been reached by the (μ+1) GA with $p_{\mathrm{c}} = 1$, the population diversity can emerge naturally from the interaction between crossover and mutation. Based on such a result on the population dynamics, bounds on the expected optimisation time of the function class are then deduced for two different settings of the algorithm: standard and high mutation rates.

IV. Population dynamics

Assume that the algorithm has reached a population where all individuals are identical and on the plateau, i.e., the less diverse setting. We refer to identical individuals as a *species*, hence, in this case, there is only one species. Eventually, a mutation will create a different search point on the plateau, leading to the creation of a new species. Both species may shrink or grow in size, and there is a chance that the new species will disappear and that we return to one species only.

However, the existence of two species also serves as a catalyst for creating further species in the following sense. Say two parents 0001111111 and 0010111111 are recombined, then crossover has a good chance of creating an individual with $n-k+1$ 1s, e.g., 0011111111. Then mutation has a constant probability of flipping any of the $n-k-1$ unrelated 1-bits to 0, leading to a new species, e. g., 0011111011. This may lead to a sudden burst of diversity in the population.

To further investigate these dynamics, we set up a preliminary experiment for $n = 500$, $k = 3$, with population size $\mu = 100$ and mutation parameter χ from $[0.1, 0.2, \dots 2.0]$. Since we are only interested in the dynamics on the plateau, the optimum is always rejected and the population is initialised with copies of a single plateau solution. 100 independent runs are repeated for each setting, and as an indicator of diversity, the size of the largest species is recorded for the first 10^5 iterations of each run. Figure 2 illustrates the obtained result. Clearly, we see that new species can emerge from time to time and more importantly if the mutation rate χ/n is sufficiently large then a diverse population can be maintained (size of the largest species remains close to 1) after some time.

The above simulation indicates that the mutation rate and the size of the largest species are important factors for describing the population diversity. With a large enough mutation rate, the size of the largest species can perform a random walk biased towards a reduction of its value. Once its size has decreased significantly from its maximum μ, there is a good chance for recombining two parents from different species. This helps in finding the global optimum, as crossover can increase the number of 1s in the offspring, compared to its parents, such that fewer bits need to be flipped by mutation to reach the optimum. This is formalised in the following lemma.

Lemma 3. *The probability that the global optimum is constructed by a uniform crossover of two parents on the plateau having Hamming distance* $2d$*, followed by mutation, is*

$$\sum_{i=0}^{2d} \binom{2d}{i} \frac{1}{2^{2d} n^{k+d-i}} \left(1 - \frac{1}{n}\right)^{n-k-d+i} \quad (4)$$

$$\geq \frac{1}{2^{2d} n^{k-d}} \left(1 - \frac{1}{n}\right)^{n-k+d}. \quad (5)$$

Proof. For a pair of search points on the plateau with Hamming distance $2d$, both parents have d 1s among the $2d$ bits that differ between parents, and $n-k-d$ 1s outside this area. Assume that crossover sets i out of these $2d$ bits to 1, which happens with probability $\binom{2d}{i} \cdot 2^{-2d}$. Then mutation needs to flip the remaining $k+d-i$ 0s to 1. The probability that such a pair creates the optimum is hence

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

The second bound is obtained by ignoring summands $i < 2d$ for the inner sum. □

Note that even a Hamming distance of 2, i.e., $d = 1$, leads to a probability of $\Omega(n^{-k+1})$, provided that such parents are selected for reproduction. The probability is by a factor of n larger than the probability $\Theta(n^{-k})$ of mutation without crossover reaching the optimum from the plateau.

We will show that this effect leads to a speedup of nearly n for the (μ+1) GA, compared to the expected time of $\Theta(n^k)$ for the (1+1) EA [10] and other EAs only using mutation.

The idea behind the analysis is to investigate the random walk underlying the size of the largest species. We bound the expected time for this size to decrease to $\mu/2$ and then argue that the (μ+1) GA is likely to spend a good amount of time with a population of good diversity, where the probability of creating the optimum in every generation is $\Omega(n^{-k+1})$ due to the chance of recombining parents of Hamming distance at least 2.

In the following, we refer to $Y(t)$ as the size of the largest species in the population at time t. Define

$$p_+(y) := \Pr(Y(t+1) - Y(t) = 1 \mid Y(t) = y)\ ,$$
$$p_-(y) := \Pr(Y(t+1) - Y(t) = -1 \mid Y(t) = y)\ ,$$

i.e., $p_+(y)$ is the probability that the size of the largest species increases from y to $y+1$, and $p_-(y)$ is the probability that it decreases from y to $y-1$.

The following lemma gives bounds on these transition probabilities, unless two parents of Hamming distance larger

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

than 2 are selected for recombination (this case will be treated later in Lemma 5). We formulate the lemma for arbitrary mutation rates $\chi/n = \Theta(1/n)$ and restrict our attention to sizes $Y(t) \geq \mu/2$ as we are only interested in the expected time for the size to decrease to $\mu/2$.

Lemma 4. *For every population on the plateau of* Jump_k *for* $k = \mathrm{o}(n)$, *the following holds. Either the (μ+1) GA with mutation rate* $\chi/n = \Theta(1/n)$ *performs a crossover of two parents whose Hamming distance is larger than 2, or the size* $Y(t)$ *of the largest species changes according to transition probabilities* $p_-(\mu) = \Omega(k/n)$ *and, for* $\mu/2 \leq y < \mu$,

$$p_+(y) \leq \frac{y(\mu-y)(\mu+y)}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n + \mathrm{O}\left(\frac{(\mu-y)^2}{\mu^2 n}\right),$$
$$p_-(y) \geq \frac{y(\mu-y)(\mu+\chi y)}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n.$$

Proof. We call an individual belonging to the current largest species a y individual and all the others non-y individuals. In each generation, there is either no change, or one individual is added to the population and one individual chosen uniformly at random is removed from the population. In order to increase the number of y individuals, it is necessary that a y individual is added to the population and a non-y individual is removed from the population. Analogously, in order to decrease the number of y individuals, it is necessary that a non-y individual is added to the population and a y individual is removed from the population.

Given that $Y(t) = y$, let $p(y)$ be the probability that a y individual is created at time $t+1$, and $q(y)$ the probability that a non-y individual is created. Since all considered individuals are on the plateau, the individual for deletion is selected uniformly at random. Multiplying by the survival probabilities we have

$$p_-(y) = q(y)\left(\frac{y}{\mu+1}\right) \text{ and} \tag{6}$$
$$p_+(y) := p(y)\left(1-\frac{y+1}{\mu+1}\right) = p(y)\left(\frac{\mu-y}{\mu+1}\right). \tag{7}$$

We now estimate an upper bound on $p(y)$. We may assume that the Hamming distance between parents is at most 2 as otherwise there is nothing to prove. A y individual can be created in the following three ways:

- Two y individuals are selected. Crossing over two y individuals produces another y individual, which survives mutation if no bits are flipped, i.e., with probability $(1-\chi/n)^n$.
- One y individual and one non-y individual are selected. The crossover operator produces a y individual with probability $1/4$ (as the individuals have Hamming distance 2 by assumption), and mutation does not flip any bits with probability $(1-\chi/n)^n$. If the crossover operator does not produce a y individual, then, to produce a y individual, at least one specific bit-position must be mutated, which occurs with probability $\mathrm{O}(1/n)$. The overall probability is hence $(1/4)(1-\chi/n)^n + \mathrm{O}(1/n)$.
- Two non-y individuals are selected. These two individuals are either identical or have Hamming distance 2 (i.e., by assumption). In the first case, they both have one of the k 0-bit positions of a y individual set to 1. In the second case, they either both have one of the k 0-bit positions of a y individual set to 1, or they both have one of the $n-k$ 1-bit positions set to 0. In both cases, crossover cannot change the value of such a bit. Thus, at least one specific bit-position must be flipped, which occurs with probability $\mathrm{O}(1/n)$.

Taking into account the probabilities of the three selection events above, the probability of producing a y individual is

$$\begin{aligned}
p(y) &= \left(\frac{y}{\mu}\right)^2\left(1-\frac{\chi}{n}\right)^n + 2\left(\frac{y}{\mu}\right)\left(1-\frac{y}{\mu}\right)\cdot \\
&\quad \cdot\left[\left(\frac{1}{4}\right)\left(1-\frac{\chi}{n}\right)^n + \mathrm{O}\left(\frac{1}{n}\right)\right] + \frac{(\mu-y)^2}{\mu^2}\mathrm{O}\left(\frac{1}{n}\right) \\
&= \left(1-\frac{\chi}{n}\right)^n\left(\frac{y}{\mu}\right)\left(\frac{y}{\mu}+\frac{\mu-y}{2\mu}\right) + \\
&\quad + \mathrm{O}\left(\frac{y(\mu-y)}{\mu^2}\cdot\frac{1}{n}\right) + \mathrm{O}\left(\frac{(\mu-y)^2}{\mu^2}\cdot\frac{1}{n}\right) \\
&= \frac{y(\mu+y)}{2\mu^2}\left(1-\frac{\chi}{n}\right)^n + \mathrm{O}\left(\frac{\mu-y}{\mu}\cdot\frac{1}{n}\right).
\end{aligned}$$

We then estimate a lower bound on $q(y)$. In the case where $y = \mu$, a non-y individual can be added to the population if

- two y individuals are selected and the mutation operator flips one of the k 0-bits and one of the $n-k$ 1-bits. This event occurs with probability

$$q(\mu) = k(n-k)\left(\frac{\chi}{n}\right)^2\left(1-\frac{\chi}{n}\right)^{n-2} \tag{8}$$
$$= \Omega\left(\frac{k}{n}-\frac{k^2}{n^2}\right) = \Omega\left(\frac{k}{n}\right), \tag{9}$$

where we used that $k = \mathrm{o}(n)$ in the last equality.

In the other case, where $y < \mu$, a non-y individual can be added to the population in the following two ways:

- A y individual and a non-y individual are selected. Crossover produces a copy of the non-y individual with probability $1/4$, which is unchanged by mutation with probability $(1-\chi/n)^n$. Secondly, with probability $1/4$, crossover produces an individual with $k-1$ 0-bits. Mutation then creates a non-y individual by flipping a single of the $n-k$ 1-bit positions that do not lead to re-creating y. Thirdly, again with probability $1/4$, crossover produces an individual with $k+1$ 0-bits and mutation then creates a non-y individual by flipping a single of k 1-bits that do not lead back to y. The above three events, conditional on selecting a y individual and a non-y individual, lead to a total probability of

$$\begin{aligned}
&\frac{1}{4}\cdot\left(1-\frac{\chi}{n}\right)^n + \frac{1}{4}\cdot(n-k)\cdot\frac{\chi}{n}\left(1-\frac{\chi}{n}\right)^{n-1} \\
&\quad + \frac{1}{4}\cdot k\cdot\frac{\chi}{n}\left(1-\frac{\chi}{n}\right)^{n-1} \\
&\geq \frac{\chi+1}{4}\cdot\left(1-\frac{\chi}{n}\right)^n.
\end{aligned}$$

- Two non-y individuals are selected. In the worst case, the selected individuals are different, hence, crossover produces an individual on the plateau with probability at least $1/2$, which mutation does not destroy with probability $(1-\chi/n)^n$.

Assuming that $\mu/2 \leq y < \mu$ and n is sufficiently large, the probability of adding a non-y individual is

$$\begin{aligned} q(y) &\geq 2\left(\frac{y}{\mu}\right)\left(1-\frac{y}{\mu}\right)\cdot\frac{\chi+1}{4}\left(1-\frac{\chi}{n}\right)^n \\ &\quad+\frac{1}{2}\left(1-\frac{y}{\mu}\right)^2\left(1-\frac{\chi}{n}\right)^n \\ &=\frac{(\mu-y)(\mu+\chi y)}{2\mu^2}\left(1-\frac{\chi}{n}\right)^n . \end{aligned}$$

Plugging $p(y)$ and $q(y)$ into equations (6) and (7), we get

$$\begin{aligned} p_-(y) &\geq \left[\frac{(\mu-y)(\mu+\chi y)}{2\mu^2}\left(1-\frac{\chi}{n}\right)^n\right]\left(\frac{y}{\mu+1}\right) \\ &=\frac{(\mu-y)(\mu+\chi y)y}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n . \end{aligned}$$

And we also have

$$\begin{aligned} p_+(y) &= \left[\frac{y(\mu+y)}{2\mu^2}\left(1-\frac{\chi}{n}\right)^n+\mathrm{O}\left(\frac{\mu-y}{\mu}\cdot\frac{1}{n}\right)\right]\left(\frac{\mu-y}{\mu+1}\right) \\ &=\frac{(\mu^2-y^2)y}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n+\mathrm{O}\left(\frac{(\mu-y)^2}{\mu^2 n}\right) . \end{aligned}$$ □

Steps where crossover recombines two parents with larger Hamming distance were excluded from Lemma 4 as they require different arguments. The following lemma shows that conditional transition probabilities in this case are favourable in that the size of the largest species is more likely to decrease than to increase.

Lemma 5. *Assume that $y \geq \mu/2$ and that the (μ+1) GA on Jump_k with $k = \mathrm{o}(n)$ and mutation rate $\chi/n = \Theta(1/n)$ selects two individuals on the plateau with Hamming distance larger than 2, then for conditional transition probabilities $p^*_-(y)$ and $p^*_+(y)$ for decreasing or increasing the size of the largest species, $p^*_-(y) \geq 2p^*_+(y)$.*

Proof. Assume that the population contains two individuals x and z with Hamming distance $2\ell \leq 2k$, where $\ell \geq 2$. Without loss of generality, let us assume that they differ in the first 2ℓ bit positions.

First assume that the individual y representing the largest species has ℓ 0-bits in the first 2ℓ positions. Then a y individual may be produced by creating the ℓ 0-bits and ℓ 1-bits in the exact positions by crossover and no followed mutation. Alternatively, at least 1 exact bit has to be flipped by mutation. Then the probability of producing a y individual from x and z and replacing a non-y individual with y is less than

$$\begin{aligned} p^*_+(y) &\leq \left[\left(\frac{1}{2}\right)^{2\ell}\left(1-\frac{\chi}{n}\right)^n+\mathrm{O}\left(\frac{1}{n}\right)\right]\left(\frac{\mu-y}{\mu}\right) \\ &\leq \left(\frac{1}{2}\right)^{2\ell+1}\left(1-\frac{\chi}{n}\right)^n+\mathrm{O}\left(\frac{1}{n}\right) . \end{aligned}$$

On the other hand, the probability of producing an individual on the plateau different from y and replacing a y individual is at least

$$\begin{aligned} p^*_-(y) &\geq \left(\binom{2\ell}{\ell}-1\right)\left(\frac{1}{2}\right)^{2\ell}\left(1-\frac{\chi}{n}\right)^n\left(\frac{y}{\mu}\right) \\ &\geq 3\left(\frac{1}{2}\right)^{2\ell+1}\left(1-\frac{\chi}{n}\right)^n \geq 2p^*_+(y) \end{aligned}$$

for sufficiently large n.

In the other case, assume that the individual y does not have ℓ 0-bits in the first 2ℓ bit-positions. Then the mutation operator must flip at least one specific bit among the last $n-2\ell$ positions to produce y, which occurs with probability $\mathrm{O}(1/n)$. The probability to produce a non-y individual on the plateau is lower bounded by the probability of the event that recombining x and z produces a bitstring with exactly k 0-bits in the first 2ℓ bit-positions, none of the bits are mutated, and a majority individual is replaced, i.e.,

$$\begin{aligned} p^*_-(y) &\geq \binom{2k}{k}2^{-2k}\left(1-\frac{\chi}{n}\right)^n\left(\frac{y}{\mu}\right) \\ &\geq \frac{2^{2k-1}}{\sqrt{k}}2^{-2k}\left(1-\frac{\chi}{n}\right)^n\left(\frac{y}{\mu}\right) = \Omega(1/\sqrt{k}). \end{aligned}$$

where the inequality follows by Stirling's inequality. Taking into account the assumption $k = \mathrm{o}(n)$, it holds for sufficiently large n that $p^*_-(y) \geq 2p^*_+(y)$. □

V. Standard mutation rate

We first analyse the (μ+1) GA with the standard mutation rate of $1/n$, i.e., $\chi = 1$. We show that the diversity emerging in the (μ+1) GA leads to a speedup of nearly n for the (μ+1) GA, compared to the expected time of $\Theta(n^k)$ for the (1+1) EA [10] and other EAs only using mutation.

Theorem 6. *The expected optimisation time of the (μ+1) GA with $p_c = 1$ and $\mu \leq \kappa n$, for some constant $\kappa > 0$, on Jump_k, $k = \mathrm{o}(n)$, is*

$$\mathrm{O}\left(\mu n\sqrt{k}\log(\mu)+n^k/\mu+n^{k-1}\log(\mu)\right) .$$

For $k \geq 3$, the best speedup is of order $\Omega(n/\log n)$ for $\mu = \kappa n$. For $k = 2$, the best speedup is of order $\Omega(\sqrt{n/\log n})$ for $\mu = \Theta(\sqrt{n/\log n})$.

Note that for mutation rate $1/n$, the dominant terms in Lemma 4 are equal, hence the size of the largest species performs a fair random walk up to a bias resulting from small-order terms. This confirms our intuition from observing simulations. The following lemma formalises this fact: in steps where the size $Y(t)$ of the largest species changes, an almost fair random walk is performed.

Lemma 7. *For the random walk induced by the size of the largest species, conditional on the current size y changing, for $\mu/2 < y < \mu$, the probability of increasing y is at most $1/2 + \mathrm{O}(1/n)$, and the probability of decreasing it is at least $1/2 - \mathrm{O}(1/n)$.*

Proof. We only have to estimate the conditional probability of increasing y as the two probabilities sum up to 1. The sought probability is given by $p_+(y)/(p_+(y)+p_-(y))$, which is strictly increasing in $p_+(y)$. Lemma 5 states that whenever the (μ+1) GA recombines two parents of Hamming distance larger than 2, the claim on conditional probabilities clearly follows. Hence we assume in the following that this does not happen.

Using the lower bound for $p_+(y)$ and the upper bound for $p_-(y)$ from Lemma 4, with implicit constant c_+ in the asymptotic term for p_+, we get

$$\begin{aligned}\frac{p_+(y)}{p_+(y)+p_-(y)} &\leq \frac{\frac{y(\mu+y)(\mu-y)}{2\mu^2(\mu+1)}\cdot\left(1-\frac{1}{n}\right)^n + \frac{c_+(\mu-y)^2}{\mu^2 n}}{\frac{y(\mu+y)(\mu-y)}{\mu^2(\mu+1)}\cdot\left(1-\frac{1}{n}\right)^n + \frac{c_+(\mu-y)^2}{\mu^2 n}}\\ &= \frac{1}{2} + \frac{\frac{c_+(\mu-y)^2}{2\mu^2 n}}{\frac{y(\mu+y)(\mu-y)}{\mu^2(\mu+1)}\cdot\left(1-\frac{1}{n}\right)^n + \frac{c_+(\mu-y)^2}{\mu^2 n}}\\ &= \frac{1}{2} + \frac{\frac{c_+(\mu-y)}{2\mu n}}{\frac{y(\mu+y)}{\mu(\mu+1)}\cdot\left(1-\frac{1}{n}\right)^n + \frac{c_+(\mu-y)}{\mu n}}\ ,\end{aligned}$$

where in the last step we multiplied the last fraction by $\mu/(\mu-y)$. Now the numerator is $\mathrm{O}(1/n)$. Since $\mu/2 < y < \mu$, we have $\frac{y(\mu+y)}{\mu(\mu+1)} = \Theta(1)$. Along with $\left(1-\frac{1}{n}\right)^n = \Theta(1)$ and $\frac{c_+(\mu-y)}{\mu n} = \mathrm{O}(1/n)$, the denominator simplifies to $\Theta(1) + \mathrm{O}(1/n) = \Theta(1)$. Hence the last fraction is $\mathrm{O}(1/n)$, proving the claim. □

We use these transition probabilities to bound the expected time for the random walk to hit $\mu/2$.

Lemma 8. *Consider the random walk of $Y(t)$, starting in state $X_0 \geq \mu/2$. Let T be the first hitting time of state $\mu/2$. If $\mu = \mathrm{O}(n)$, then $\mathrm{E}(T \mid X_0) = \mathrm{O}\big(\mu n + \mu^2\log\mu\big)$ regardless of X_0.*

Proof. Let E_i abbreviate $\mathrm{E}(T \mid X_0 = i)$, then $E_{\mu/2} = 0$. Since $p_-(\mu) = \Omega(1/n)$ by Lemma 4, the expected time to leave state μ towards state $\mu-1$ is $1/p_-(\mu) = \mathrm{O}(n)$ and the remaining time will be $E_{\mu-1}$, thus $E_\mu = \mathrm{O}(n) + E_{\mu-1}$.

For $\mu/2 < y < \mu$, the probability of leaving state y is always (regardless of Hamming distances between species) bounded from below by the probability of selecting two y individuals as parents, not flipping any bits during mutation, and choosing a non-y individual for replacement:

$$p_+(y)+p_-(y) \geq \frac{y^2}{\mu^2}\cdot\left(1-\frac{1}{n}\right)^n\cdot\frac{\mu-y}{\mu+1} \geq \frac{\mu-y}{24\mu}\ ,$$

as $y \geq \mu/2$, $\mu+1 \leq 3\mu/2$ (since $\mu \geq 2$), and $(1-1/n)^n \geq 1/4$ for $n \geq 2$. Hence the expected time for leaving state i towards either state $i+1$ or state $i-1$ is at most $24\mu/(\mu-i)$. Using conditional transition probabilities $1/2 \pm \delta$ for $\delta = \mathrm{O}(1/n)$ according to Lemma 7, E_i is bounded as

$$E_i \leq \frac{24\mu}{\mu-i} + \left(\frac{1}{2}-\delta\right)E_{i-1} + \left(\frac{1}{2}+\delta\right)E_{i+1}\ .$$

This is equivalent to

$$\left(\frac{1}{2}-\delta\right)\cdot(E_i - E_{i-1}) \leq \frac{24\mu}{\mu-i} + \left(\frac{1}{2}+\delta\right)\cdot(E_{i+1}-E_i)\ .$$

Introducing $D_i := E_i - E_{i-1}$, this is

$$\left(\frac{1}{2}-\delta\right)\cdot D_i \leq \frac{24\mu}{\mu-i} + \left(\frac{1}{2}+\delta\right)\cdot D_{i+1}$$

and equivalently

$$D_i \leq \frac{\frac{24\mu}{\mu-i} + \left(\frac{1}{2}+\delta\right)\cdot D_{i+1}}{\frac{1}{2}-\delta} \leq \frac{50\mu}{\mu-i} + \alpha\cdot D_{i+1}$$

for $\alpha := \frac{1+2\delta}{1-2\delta} = 1 + \mathrm{O}(1/n)$, assuming n is large enough. From $E_\mu = \mathrm{O}(n) + E_{\mu-1}$, we get $D_\mu = \mathrm{O}(n)$, hence an induction yields

$$D_i \leq \sum_{j=i}^{\mu-1}\frac{50\mu}{\mu-j}\cdot\alpha^{j-i} + \alpha^{\mu-i}\cdot\mathrm{O}(n)\ .$$

Combining $\alpha = 1 + \mathrm{O}(1/n)$ and $1 + x \leq \mathrm{e}^x$ for all $x \in \mathbb{R}$, we have $\alpha^\mu \leq \mathrm{e}^{\mathrm{O}(\mu/n)} \leq \mathrm{e}^{\mathrm{O}(1)} = \mathrm{O}(1)$. Bounding both α^{j-i} and $\alpha^{\mu-i}$ in this way, we get

$$D_i \leq \mathrm{O}(n) + \mathrm{O}(\mu)\cdot\sum_{j=i}^{\mu-1}\frac{1}{\mu-j} = \mathrm{O}(n+\mu\log\mu)\ ,$$

as the sum is equal to $\sum_{j=1}^{\mu-i} 1/j = \mathrm{O}(\log\mu)$.

Now,

$$\begin{aligned}&D_{\mu/2+1} + D_{\mu/2+2} + \dots + D_i\\ &\quad= (E_{\mu/2+1} - E_{\mu/2}) + (E_{\mu/2+2} - E_{\mu/2+1}) + \dots\\ &\qquad+ (E_i - E_{i-1})\\ &\quad= E_i - E_{\mu/2} = E_i\ .\end{aligned}$$

Hence, we get $E_i = \sum_{k=\mu/2+1}^{i} D_k \leq \mathrm{O}\big(\mu n + \mu^2\log\mu\big)$. □

Now we show that when the largest species has decreased its size to $\mu/2$ there is a good chance that the optimum will be found within the following $\Theta(\mu^2)$ generations.

Lemma 9. *Consider the (μ+1) GA with $p_c = 1$ on Jump_k. If the largest species has size at most $\mu/2$ and $\mu \leq \kappa n$ for a small enough constant $\kappa > 0$, the probability that during the next $c\mu^2$ generations, for some constant $c > 0$, the global optimum is found is $\Omega\Big(\frac{1}{1+n^{k-1}/\mu^2}\Big)$.*

Proof. We show that during the $c\mu^2$ generations the size of the largest species never rises above $(3/4)\mu$ with at least constant probability. Then we calculate the probability of jumping to the optimum during the phase given that this happens.

Let X_i, $1 \leq i \leq c\mu^2$ be random variables indicating the change in the number of individuals of the largest species at generation i. We pessimistically ignore self-loops and assume that the size of the species either increases or decreases in each generation, thus $X_i \in \{-1, +1\}$. Using the conditional probabilities from Lemma 7, we get that the expected increase in each step is

$$1\cdot(1/2 + \mathrm{O}(1/n)) - 1\cdot(1/2 - \mathrm{O}(1/n)) = \mathrm{O}(1/n)\ .$$

Then the expected increase in size of the largest species at the end of the phase is

$$\mathrm{E}(X) = \sum_{i=1}^{c\mu^2} X_i = \sum_{i=1}^{c\mu^2}\mathrm{O}(1/n) = (c'\mu^2)/n \leq c'\kappa\mu \leq \mu/8,$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

where we use that $\mu \leq \kappa n$ and κ is chosen small enough.

Using a Hoeffding bound, we get $\Pr(X \geq \mathrm{E}(X)+\lambda) \leq \exp(-2\lambda^2/\sum_{i=1}^{c\mu^2} c_i^2)$. We then use that $\lambda = \mu/8$ and $c_i = 2$ (i.e. the length of the interval in which X_i lives), which gives $\Pr(X \geq (2/8)\mu) \leq \exp(-c') = 1-\Omega(1)$ for some constant $c' > 0$. We remark that the bounds also hold for any partial sum of the sequence $X_1, \dots, X_{c\mu^2}$ ([1], Chapter 1, Theorem 1.13), i.e., with probability $\Omega(1)$ the size *never* exceeds $(3/4)\mu$ in the considered phase of length $c\mu^2$ generations.

While the size does not exceed $(3/4)\mu$, in every step there is a probability of at least $1/4 \cdot 3/4 = \Omega(1)$ of selecting parents from two different species. As these have Hamming distance $2d$ for some $d \geq 1$, by Lemma 3, the probability of creating the optimum is at least $2^{-2d} n^{-k+d}(1-1/n)^{n-k+d} \geq \Omega(n^{-k+1})$ for any $d \geq 1$.

Finally, the probability that at least one successful generation occurs in a phase of $c\mu^2$ is, using $1-(1-p)^\lambda \geq (\lambda p/(1+\lambda p))$ for $\lambda \in \mathbb{N}$, $p \in [0,1]$ [2, Lemma 10], the probability that the optimum is found in one of these steps is

$$1-\left(1-\frac{1}{\Omega(n^{-k+1})}\right)^{c\mu^2} \geq \Omega\left(\frac{\mu^2 \cdot n^{-k+1}}{1+\mu^2 \cdot n^{-k+1}}\right). \quad \square$$

Finally, we assemble all lemmas to prove our main theorem of this section.

Proof of Theorem 6. The expected time for the whole population to reach the plateau is $\mathrm{O}\left(\mu n\sqrt{k}\log(\mu) + n\sqrt{k}\log n\right)$ by Lemma 2.

Once the population is on the plateau, we wait till the largest species has decreased its size to at most $\mu/2$. According to Lemma 8, the time for the largest species to reach size $\mu/2$ is $\mathrm{O}(\mu n + \mu^2 \log \mu)$. By Lemma 9, the probability that in the next $c\mu^2$ steps the optimum is found is $\Omega\left(\frac{1}{1+n^{k-1}/\mu^2}\right)$. If not, we repeat the argument. The expected number of such trials is $\mathrm{O}(1+n^{k-1}/\mu^2)$, and the expected length of one trial is $\mathrm{O}(\mu n + \mu^2 \log \mu) + c\mu^2 = \mathrm{O}(\mu n + \mu^2 \log \mu)$. The expected time for reaching the optimum from the plateau is hence at most $\mathrm{O}(\mu n + \mu^2 \log(\mu) + n^k/\mu + n^{k-1}\log(\mu))$.

Adding up all times and subsuming terms $\mu^2 \log(\mu) = \mathrm{O}\left(\mu n \sqrt{k}\log \mu\right)$ and $n\sqrt{k}\log n = \mathrm{O}(n^k/\mu + n^{k-1}\log \mu)$, noting that $k = \mathrm{o}(n)$ completes the proof. $\square$

VI. High mutation rates

We now consider the run time of (μ+1) GA with mutation rate $\chi/n = (1+\delta)/n$ for an arbitrary constant $\delta > 0$. The following theorem states that in this setting the algorithm has at least a linear speedup compared to the (μ+1) EA without crossover [34]. By assuming a slightly higher mutation rate, we not only obtain a bound which is by a log-factor better than Theorem 6, but the analysis is also significantly simpler.

Theorem 10. *The (μ+1) GA with mutation rate $(1+\delta)/n$, for a constant $\delta > 0$, and population size $\mu \geq ck\ln(n)$ for a sufficiently large constant $c > 0$, has for $k = \mathrm{o}(n)$ expected optimisation time $\mathrm{O}\left(n\sqrt{k}\mu\log(\mu) + \mu^2 + n^{k-1}\right)$ on Jump_k.*

We again study the random walk corresponding to the size of the largest species on the plateau. For mutation rate $1/n$, this is almost an unbiased random walk. For slightly higher mutation rates, we will see that the random walk changes to an unfair random walk where the size of the largest species decreases by $\Omega(1/\mu)$ in expectation. Formally, our analysis assumes the following condition.

Condition 11. *For a constant $\delta > 0$ and all $y, \mu/2 \leq y \leq \mu$,*

$$p_-(y) \geq \begin{cases} \Omega(1/n) & \text{if } y = \mu, \\ \Omega(1/\mu) & \text{if } \mu/2 \leq y < \mu, \text{ and} \\ (1+\delta)p_+(y) & \text{if } \mu/2 \leq y < \mu. \end{cases} \tag{10}$$

The following lemma states that it is sufficient to increase the mutation rate slightly above $1/n$ to satisfy the diversity condition.

Lemma 12. *If $\chi/n \geq (1+\delta)/n$ for any constant $\delta > 0$, then Condition 11 holds.*

Proof. The first two inequalities follow directly from Lemma 4 and Lemma 5. For any constant $\varepsilon > 0$, Lemma 4 implies that

$$p_+(y) \leq \frac{y(\mu-y)(\mu+y)(1+\varepsilon)}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n \text{ and}$$
$$p_-(y) \geq \frac{y(\mu-y)(\mu+\chi y)(1-\varepsilon^2)}{2\mu^2(\mu+1)}\left(1-\frac{\chi}{n}\right)^n.$$

Thus, given that $\mu/2 < y < \mu$ and $\chi \geq 1+\delta$,

$$\frac{p_-(y)}{p_+(y)} \geq \left(\frac{\mu+\chi y}{\mu+y}\right)(1-\varepsilon) \geq 1+\delta'$$

for some constant $\delta' > 0$ when ε is sufficiently small. $\square$

Given Condition 11, the additive drift theorem [16] implies that the largest species quickly decreases to half the population size.

Lemma 13. *If Condition 11 holds, then the expected time until the largest species has size at most $\mu/2$ is $\mathrm{O}(\mu^2+n)$.*

Proof. Let $Y(t)$ denote the size of the largest species at time t. We consider the drift with respect to the distance function

$$h(y) := f(y) + g(y)$$

which has the two terms

$$f(y) := y, \text{ and}$$
$$g(y) := (n/\mu)\mathrm{e}^{-\kappa(\mu-y)}$$

with $\kappa := \ln(1+\delta)$ over the interval $y \in [\mu/2, \mu]$. Due to linearity of expectation, we can consider the drift of the two terms $f(y)$ and $g(y)$ separately. The second term $g(y)$ is introduced to handle the case $y = \mu$ and is defined exponentially decreasing in $\mu - y$ to avoid negative drift in the case $y = \mu - 1$. The total distance is $h(\mu) - h(\mu/2) = \mathrm{O}(\mu + n/\mu)$, hence, we need to prove that the drift of the process $h(Y(t))$ is $\Omega(1/\mu)$.

We first consider the drift with respect to the first term $f(y) = y$.

Case 1: $Y(t) = \mu$. Since $Y(t+1) \leq \mu$ for all t, the drift in this case is

$$\begin{aligned}\mathrm{E}(f(Y(t)) - f(Y(t+1)) \mid Y(t) = \mu) \\ = \mathrm{E}(\mu - Y(t+1) \mid Y(t) = \mu) \geq 0.\end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

Case 2: $\mu/2 < Y(t) < \mu$. By (10), the drift in this case is

$$\begin{aligned}&\mathrm{E}(f(Y(t)) - f(Y(t+1)) \mid Y(t) = y, \mu/2 < y < \mu)\\ &\quad = p_-(y)(y - (y-1)) + p_+(y)(y - (y+1))\\ &\qquad\qquad = p_-(y) - p_+(y) > \delta p_-(y) = \Omega(1/\mu)\ .\end{aligned}$$

We then consider the drift with respect to the second term $g(y) = (n/\mu)e^{-\kappa(\mu-y)}$.

Case 1: $Y(t) = \mu$. By (10),

$$\begin{aligned}&\mathrm{E}(g(Y(t)) - g(Y(t+1)) \mid Y_t(t) = \mu)\\ &\qquad\qquad = \Omega(1/n)(n/\mu)(1 - \mathrm{e}^{-\kappa}) = \Omega(1/\mu)\ .\end{aligned}$$

Case 2: $\mu/2 < Y(t) < \mu$. By (10), $p_+(y)\mathrm{e}^{\kappa} \leq p_-(y)$. The drift with respect to g is therefore

$$\begin{aligned}&\mathrm{E}(g(Y(t)) - g(Y(t+1)) \mid \mu/2 < Y(t) < \mu)\\ &= p_+(y)(g(y) - g(y+1)) + p_-(y)(g(y) - g(y-1))\\ &= (n/\mu)\mathrm{e}^{-\kappa(\mu-y+1)}(\mathrm{e}^{\kappa} - 1)(p_-(y) - p_+(y)\mathrm{e}^{\kappa}) > 0\ .\end{aligned}$$

To complete the proof, we now consider the drift of the overall distance function h. In both Case 1 and Case 2, it holds that

$$\begin{aligned}&\mathrm{E}(h(Y(t)) - h(Y(t+1)) \mid Y_t(t)) =\\ &\qquad \mathrm{E}(f(Y(t)) - f(Y(t+1)) \mid Y_t(t))\\ &\qquad + \mathrm{E}(g(Y(t)) - g(Y(t+1)) \mid Y_t(t)) = \Omega(1/\mu),\end{aligned}$$

and the theorem follows. □

After the population diversity has increased sufficiently on the plateau, an optimal solution can be produced with the right combination of crossover and mutation. This is captured by the following lemma.

Lemma 14. *Consider a population P on the Jump_k plateau ($f(x) = n-k$ for all $x \in P$). We partition P into species. For any constant $0 < c < 1$, if the largest species has size at most $c\mu$, then the optimal solution is created by uniform crossover followed by mutation with probability $\Omega((\chi/n)^{k-1})$ assuming the mutation rate is $\chi/n = \Theta(1/n)$.*

Proof. Since the size of the largest species 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 and 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

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

where we bound the sum by dropping all but the last term ($i = 2d$) and use $4^{-d} \geq \frac{1}{4}\left(\frac{\chi}{n}\right)^{d-1}$, since $d > 0$ and we take n to be large enough. □

We are now in a position to complete the run time analysis of the algorithm. By Lemma 2 and Lemma 13, we quickly reach a diverse population on the plateau. From this configuration, there is a sufficiently high probability that before the diversity is lost the algorithm has crossed over an appropriate pair of individuals and jumped to the optimum. If the diversity is lost, we can repeat the argument.

Proof of Theorem 10. By Lemma 2, the expected time for the entire population to reach the plateau is $\mathrm{O}\left(n\sqrt{k}\mu\log\mu\right)$, and by Lemma 12, Condition 11 is satisfied.

Assume c' is sufficiently large so that $\mu \geq (c'k/\delta)\ln(n)$ implies $(1+\delta)^{\mu/4} \geq 4cn^{k-1} + 1$ for a constant c that will be determined. We consider a phase of length $c(\mu^2 + 2n^{k-1})$ iterations and define the following three failure events.

The *first failure* occurs if within the first $c(\mu^2 + n)$ iterations the largest species has not become smaller than $\mu/2$ individuals. By Lemma 13, the expected time until less than $\mu/2$ individuals belong to the largest species is $\mathrm{O}(\mu^2 + n)$. Hence, by Markov's inequality, the probability of this failure is less than $1/4$ when c is sufficiently large.

The *second failure* occurs if within the next cn^{k-1} iterations there exists a sub-phase which starts with $\mu/2+1$ individuals in the largest species and ends with the largest species larger than $(3/4)\mu$ without first reducing to $\mu/2$. We call such a sub-phase a *failure*. We model the number of individuals in the largest species by a Gambler's ruin argument [12], where, by (10), the probability of losing an individual in the largest species is at least a $(1+\delta)$-factor larger than the probability of winning such an individual. From standard results about the Gambler's ruin process [12], the probability that a sub-phase is a failure is $\delta/((1+\delta)^{\mu/4} - 1)$. By a union bound, the probability that any of the at most cn^{k-1} sub-phases is a failure is no more than $cn^{k-1}/((1+\delta)^{\mu/4} - 1) < 1/4$.

The *third failure* occurs if the optimum is not found during a sub-phase of length cn^{k-1} iterations where the largest species is always smaller than $(3/4)\mu$ individuals. In this configuration, two individuals with Hamming distance at least 2 are selected with probability at least $(3/4)(1/4)$. By Lemma 14, the probability of obtaining the optimum from two such individuals is $\Omega(1/n^{k-1})$. Hence, the probability of not obtaining the optimum during the sub-phase of length cn^{k-1} is $(1 - \Omega(1/n^{k-1}))^{cn^{k-1}} \leq 1/4$ for sufficiently large c.

By a union bound, given a sufficiently large constant $c > 0$, the probability that none of the failures occur and the optimum is found within a phase of length $c(\mu^2 + 2n^{k-1})$ iterations is at least $1/4$. Therefore, the expected number of phases until the optimum is found is no more than 4. □

VII. Experiments

Since the theoretical results presented in the previous section are asymptotic and they only provide upper bounds on the run time of the algorithms, we also implemented the (μ+1) GA and conducted experiments on Jump_k for various values of k, n, and p_{m}.

In each tested setting of the algorithm and the function, the run is replicated 100 times with different random seeds. The number of function evaluations, denoted as '# evaluations', is reported as the run time. The population size is set to $\mu =$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

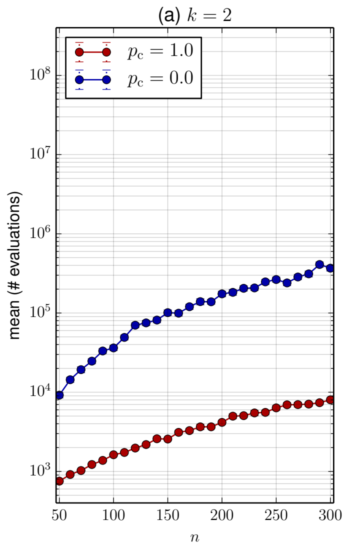

Figure 3. The impact of enabling crossover.

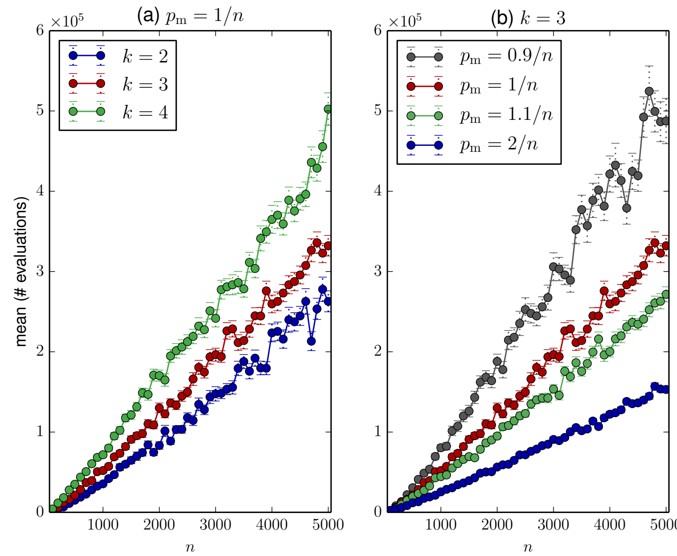

Figure 4. Run time for different jump lengths k and different mutation rates p_m with crossover.

$4e \ln n$ so that a realistic population of at least 40 individuals is always assumed (e. g., even for $n = 50$ in Figure 3).

A. Impact of crossover and mutation rates

Figure 3 (a) and (b) depict the performance of the GA ($p_c = 1.0$) compared to the algorithm using only mutation ($p_c = 0.0$) under the same setting ($p_m = 1/n$). The range of n in this experiment is set to $n = [50 \dots 300]$ with a step size of 10, and k is in $\{2, 3\}$. Even with these small values of k and n, a strong reduction of the average run time can be observed, up to a multiplicative factor of 10^4.

The impact of the jump length k on the run time is illustrated in Figure 4 (a). The experiment was set with n in $[100 \dots 5000]$ (with a step size of 100) and k is in $\{3, 4, 5\}$. We notice that the increase of k does not imply a large change in the average run time. The average run time seems to still scale linearly with n in this setting even for $k = 4$. By fixing $k = 3$, we also experimented with different mutation rates, i.e., p_m in $\{0.9/n, 1/n, 1.1/n, 2/n\}$. The results are displayed in Figure 4 (b). We notice that the mutation rates above $1/n$ reduce the average run time while a slightly lower mutation rate increases it considerably. With mutation rate $2/n$, the average run time and the stability of the runs are distinctively improved.

On the other hand, an excessive increase of the mutation rate may deteriorate the average run time because of the likelihood of multiple bit flips which imply harmful mutations. This can be observed in the experiment depicted in Figure 5 (in log-scale) for $n = 500$. In this experiment, k is in $\{2, 3, 4\}$, and the range of $\chi = p_m \cdot n$ is set to $[0.6 \dots 8]$ (with a step size of 0.1). We note that the more k is increased, the stronger the negative effect of high mutation rates can be noticed. Moreover, too low mutation rates are also bad for the run time. This can be related to our theoretical analysis, in which a low mutation rate could have made the random walk associated with the size of the largest species biased toward the wrong direction. This may lead to the reduction of the population diversity and the loss of benefit from crossover.

B. Comparison with the use of diversity mechanisms

In a previous study [7], we have shown that many common mechanisms to preserve population diversity can speed up significantly the expected optimisation time of (μ+1) GA (with standard mutation rate) on Jump_k when crossover is enabled. The aim of this section is to compare by experiments the setting of high mutation rate with the results taken directly from [7] for six[1] mechanisms: duplicate minimisation and elimination, deterministic crowding, convex hull maximisation, fitness sharing and island model.

Again full crossover is enabled ($p_c = 1.0$), but the problem size n is varied in $[100, 1000]$ (with a step size of 25). The result for $k = 4$ is shown in Figure 6 which also includes the setting of (μ+1) GA with standard mutation rate and without any diversity mechanism as a reference. Here the high mutation rate is set with $p_m = 2.6/n$ (the best choice for $n = 500$ and $k = 4$, previously suggested by Figure 5). An interesting observation from the experimental results is that it appears the setting of high mutation rate can be as efficient as the implementation of specific diversity mechanisms. Specifically, in Figure 6 the setting of high mutation rate is only worse than convex hull maximisation and fitness sharing.

VIII. Conclusion

A rigorous analysis of the (μ+1) GA has been presented showing how combining the use of crossover with that of mutation considerably speeds up the run time for Jump_k compared to algorithms using mutation only.

It is traditionally believed that crossover is useful only in the presence of sufficient diversity, and the emergence of this

[1]The maximisation of Hamming distance was also counted as a diversity mechanism in [7]. However, in that paper we have proved that under some conditions the mechanism is equivalent to fitness sharing.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2724201, IEEE Transactions on Evolutionary Computation

diversity is typically attributed to the mutation operator [11], [15], [35]. In general, the dynamics of mutation and crossover are vastly complex, and the question of how the two operators interact to balance exploration and exploitation has been open for decades [30]. Nevertheless, previous theoretical results on the benefit of crossover have relied solely on mutation for establishing the diversity necessary for recombination. For example, on the Jump_k function (with the exception of our own work in [7]), proofs have required an unrealistically small crossover probability in order to force long phases during which mutation alone builds up enough diversity before a useful crossover operation can be applied.

Diversity can also be enforced using artificial mechanisms, and such techniques lead to more efficient evolutionary algorithms both empirically [4], [32] and theoretically [13], [28]. Artificially enforced diversity can also be used in proofs that crossover is beneficial without having to rely on mutation alone to create sufficient variation [7].

The question to what degree the *interplay* between both crossover and mutation promotes the natural emergence of diversity in the population has been so far open. Our analysis shows that this interplay on the plateau of local optima of the Jump_k function quickly leads to a burst of diversity that is then exploited by both operators to reach the global optimum.

The balance between the amount of mutation and crossover impacts the run time considerably. While mutation rates lower than the standard $1/n$ rate considerably increase the expected run time, rates that are slightly higher than $1/n$ lead to improved performance. These rates also depend on the presence of crossover. For instance, for $k = 4$, the best rate for a mutation-only algorithm is $4/n$ while the best rate for the (μ+1) GA with $p_c = 1$ is considerably lower than $4/n$ and higher than $1/n$.

It is an open problem for future work whether crossover can lead to more than linear speedups on Jump_k for realistic crossover probabilities. Our analysis could be improved by taking into account crossover between plateau individuals with Hamming distance larger than 2. For large k, this could lead to super-linear speedups. In fact, our experiments reveal that the average run time of the (μ+1) GA does not increase considerably when k is increased from 2 to 4. However, completely new techniques may be required to improve our analysis. Finally, future work should address the interplay between mutation and crossover on fitness landscapes with different characteristics than the Jump_k function, such as those featuring neutral networks.

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. This research benefitted from Dagstuhl seminar 16011 ‘Evolution and Computing’ and is based upon work from COST Action CA15140 ‘Improving Applicability of Nature-Inspired Optimisation by Joining Theory and Practice (ImAppNIO)’ supported by COST (European Cooperation in Science and Technology).

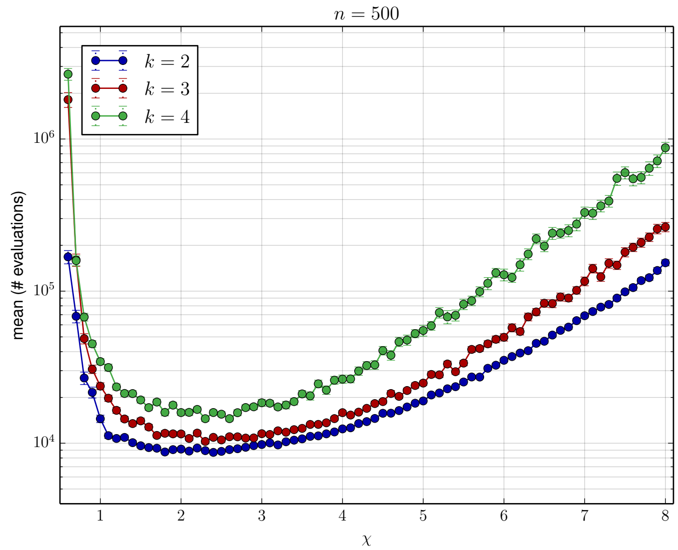

Figure 5. The impact of different mutation rates $p_m = \chi/n$ with crossover for a problem size of 500.

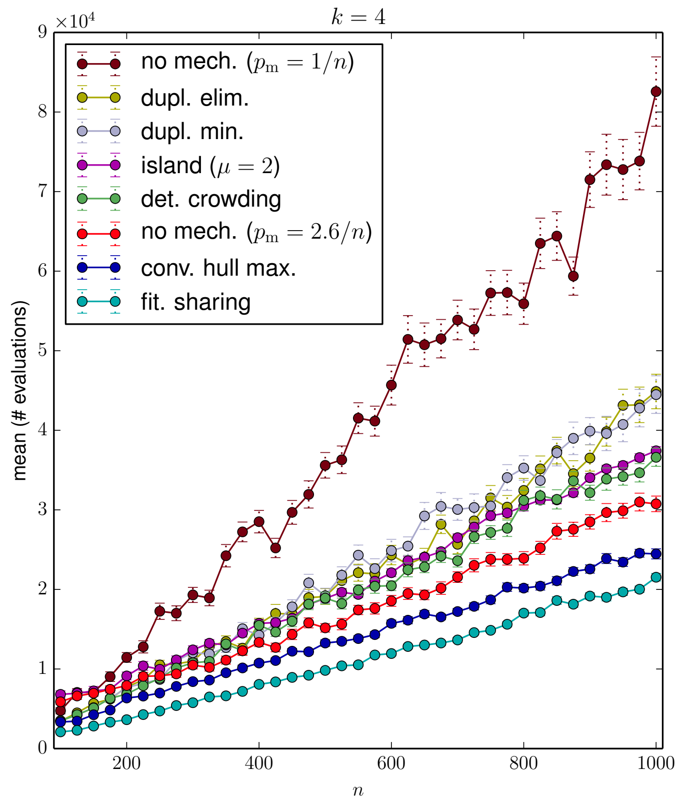

Figure 6. The performance of the diversity mechanisms for jump length 4; the mutation rate p_m is set to $1/n$ unless specified.

References

[1] A. Auger and B. Doerr, *Theory of Randomized Search Heuristics*. World Scientific, 2011.

[2] G. Badkobeh, P. K. Lehre and D. Sudholt, ‘Black-box complexity of parallel search with distributed populations’, in *Proc. of FOGA XIII*, 2015, pp. 3–15.

[3] N. Barton and T. Paixão, 'Can quantitative and population genetics help us understand evolutionary computation?', in *Proc. of GECCO '13*, 2013, pp. 1573–1580.

[4] N. Chaiyaratana, T. Piroonratana and N. Sangkawelert, 'Effects of diversity control in single-objective and multi-objective genetic algorithms', *J. Heuristics*, vol. 13, pp. 1–34, 2007.

[5] D. Corus, D.-C. Dang, A. V. Eremeev and P. K. Lehre, 'Level-based analysis of genetic algorithms and other search processes', in *Proc. of PPSN XIII*, 2014.

[6] D.-C. Dang, T. Friedrich, T. Kotzing, M. Krejca, P. K. Lehre, P. Oliveto, D. Sudholt and A. Sutton, 'Emergence of diversity and its benefits for crossover in genetic algorithms', in *Proc. of PPSN XIV*, 2016, pp. 890–900.

[7] ——, 'Escaping local optima with diversity mechanisms and crossover', in *Proc. of GECCO '16*, 2016, pp. 645–652.

[8] B. Doerr, C. Doerr and F. Ebel, 'From black-box complexity to designing new genetic algorithms', *Theor. Comput. Sci.*, vol. 567, pp. 87–104, 2015.

[9] B. Doerr, E. Happ and C. Klein, 'Crossover can provably be useful in evolutionary computation', *Theor. Comput. Sci.*, vol. 425, pp. 17–33, 2012.

[10] S. Droste, T. Jansen and I. Wegener, 'On the analysis of the $(1+1)$ Evolutionary Algorithm', *Theor. Comput. Sci.*, vol. 276, pp. 51–81, 2002.

[11] L. J. Eshelman, 'Genetic algorithms', in *Handbook of Evolutionary Computation*, Oxford University Press, 1997.

[12] W. Feller, *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, 1968.

[13] T. Friedrich, P. S. Oliveto, D. Sudholt and C. Witt, 'Analysis of diversity-preserving mechanisms for global exploration', *Evol. Comput.*, vol. 17, pp. 455–476, 2009.

[14] B. W. Goldman and W. F. Punch, 'Fast and efficient black box optimization using the parameter-less population pyramid', *Evol. Comput.*, vol. 23, pp. 451–479, 2015.

[15] *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[16] J. He and X. Yao, 'A study of drift analysis for estimating computation time of evolutionary algorithms', *Nat. Comput.*, vol. 3, pp. 21–35, 2004.

[17] T. Jansen, *Analyzing Evolutionary Algorithms – The Computer Science Perspective*. Springer, 2013.

[18] T. Jansen and I. Wegener, 'The analysis of evolutionary algorithms – a proof that crossover really can help', *Algorithmica*, vol. 34, pp. 47–66, 2002.

[19] N. L. Komarova, E. Urwin and D. Wodarz, 'Accelerated crossing of fitness valleys through division of labor and cheating in asexual populations', *Scientific Reports*, 2012.

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

[21] P. K. Lehre and X. Yao, 'Crossover can be constructive when computing unique input–output sequences', *Soft Comput.*, vol. 15, pp. 1675–1687, 2011.

[22] K.-F. Man, K.-S. Tang and S. Kwong, 'Genetic algorithms: concepts and applications', *IEEE Trans. Ind. Electron.*, vol. 43, pp. 519–534, 1996.

[23] 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*, 2011, pp. 1587–1594.

[24] F. Neumann and C. Witt, *Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity*. Springer, 2010.

[25] P. S. Oliveto, J. He and X. Yao, 'Analysis of population-based evolutionary algorithms for the vertex cover problem', in *Proc. of CEC '08*, 2008.

[26] P. S. Oliveto and C. Witt, 'On the runtime analysis of the simple genetic algorithm', *Theor. Comput. Sci.*, vol. 545, pp. 2–19, 2014.

[27] ——, 'Improved time complexity analysis of the simple genetic algorithm', *Theor. Comput. Sci.*, vol. 605, pp. 21–41, 2015.

[28] P. S. Oliveto and C. Zarges, 'Analysis of diversity mechanisms for optimisation in dynamic environments with low frequencies of change', *Theor. Comput. Sci.*, vol. 561, pp. 37–56, 2015.

[29] A. Prügel-Bennett, 'Benefits of a population: five mechanisms that advantage population-based algorithms', *IEEE Trans. Evol. Comput.*, vol. 14, pp. 500–517, 2010.

[30] W. M. Spears, 'Crossover or mutation?', in *Proc. of FOGA II*, 1992, pp. 221–237.

[31] D. Sudholt, 'Crossover speeds up building-block assembly', in *Proc. of GECCO '12*, 2012, pp. 689–696.

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

[33] D. B. Weissman, M. W. Feldman and D. S. Fisher, 'The rate of fitness-valley crossing in sexual populations', *Genetics*, vol. 186, pp. 1389–1410, 2010.

[34] C. Witt, 'Runtime analysis of the $(\mu + 1)$ EA on simple pseudo-Boolean functions', *Evol. Comput.*, vol. 14, pp. 65–86, 2006.

[35] M. Črepinšek, S.-H. Liu and M. Mernik, 'Exploration and exploitation in evolutionary algorithms: a survey', *ACM Comput. Surv.*, vol. 45, 35:1–35:33, 2013.

Duc Cuong Dang received the Engineer's degree in Computer and Network, and the MSc degree in Computer Science from the Université de Technologie de Compiègne, France in 2008. He obtained the PhD degree in Computer Science and Operational Research from the same university in 2011.

From 2011 to 2012, he was a Lecturer at the Université de Technologie de Compiègne, and from 2013 to 2016 he held a Research Fellow position at the University of Nottingham, UK. He is interested in applying Mathematics and Computer Science to solve real-world optimisation problems. His current research interests also include the theoretical aspects of heuristic and metaheuristic search.

Tobias Friedrich received the M.S. degree in computer science from the University of Sheffield, Sheffield, U.K., in 2003, the Diploma degree in mathematics from the University of Jena, Jena, Germany, in 2005, and the Ph.D. degree in computer science from Saarland University, Saarbrücken, Germany, in 2007.

He was a Post-Doctoral Fellow with the Algorithms Group, International Computer Science Institute, Berkeley, CA, USA. From 2011 to 2012, he was a Senior Researcher with the Max Planck Institute for Informatics, Saarbrücken, and an Independent Research Group Leader with the Cluster of Excellence on Multimodal Computing and Interaction, Saarbrücken. From 2012 to 2015, he was a Full Professor and the Chair of theoretical computer science with the University of Jena. Since 2015, he has been a Full Professor with the University of Potsdam, Potsdam, Germany, and the Head of the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam. His current research interests include randomised methods in mathematics and computer science and randomised algorithms (both classical and evolutionary).

Timo Kötzing received the Ph.D. degree in computer science from the University of Delaware, Newark, DE, USA, in 2009.

He was a Post-Doctoral Fellow with the Algorithms and Complexity Group, Max Planck Institute for Informatics, Saarbrücken, Germany. From 2013 to 2015, he was a Post-Doctoral Researcher with the University of Jena, Jena, Germany. Since 2015, he has been a Post-Doctoral Researcher with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany. His current research interests include theoretical foundation of learning theory as well as the analysis of randomised search heuristics and the development of efficient tools for this goal.

Martin S. Krejca received the B.S. and M.S. degrees in computer science from the University of Jena, Jena, Germany, in 2012 and 2014, respectively. He is currently pursuing the Ph.D. degree with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany.

His current research interests include the theoretical analysis of evolutionary algorithms, especially the analysis of estimation of distribution algorithms.

Per Kristian Lehre is a Senior Lecturer at the University of Birmingham, UK.

He received MSc and PhD degrees in Computer Science from the Norwegian University of Science and Technology (NTNU). After finishing his PhD in 2006, he held postdoctorial positions in the School of Computer Science at the University of Birmingham and at the Technical University of Denmark. From 2011, he was a Lecturer in the School of Computer Science at the University of Nottingham, until 2017, when he returned to Birmingham.

Dr Lehre's research interests are in theoretical aspects of nature-inspired search heuristics, in particular, runtime analysis of population-based evolutionary algorithms. His research has won several best paper awards, including at GECCO (2013, 2010, 2009, 2006), ICSTW (2008), and ISAAC (2014). He is editorial board member of Evolutionary Computation, and associate editor of IEEE Transactions on Evolutionary Computation. He was the coordinator of the successful 2M euro EU-funded project SAGE which brought together the theory of evolutionary computation and population genetics.

Pietro S. Oliveto is a Senior Lecturer and an EPSRC Early Career Fellow at the University of Sheffield, UK. He received the Laurea degree in computer science from the University of Catania, Italy in 2005 and the PhD degree from the University of Birmingham, UK in 2009. He has been EPSRC PhD+ Fellow and EPSRC Postdoctoral Fellow at Birmingham and Vice-Chancellor's Fellow at Sheffield.

His main research interest is the performance analysis of bio-inspired computation techniques including evolutionary algorithms, genetic programming, artificial immune systems and hyperheuristics. He has won best paper awards at GECCO 2008, ICARIS 2011 and GECCO 2014. He is part of the Steering Committee of the annual workshop on Theory of Randomized Search Heuristics (ThRaSH), Associate Editor of the IEEE Transactions on Evolutionary Computation, Chair of the IEEE CIS Task Force on Theoretical Foundations of Bio-inspired Computation and member of the IEEE Peer Review College.

Dirk Sudholt is a Senior Lecturer at the University of Sheffield, UK, heading the Algorithms Group.

He received his Diplom in 2004 and his PhD in computer science in 2008 from the Technische Universität Dortmund, Germany, under the supervision of Prof. Ingo Wegener. He has held postdoc positions at the International Computer Science Institute in Berkeley, CA, USA and at the University of Birmingham, UK. His research focuses on the runtime analysis of randomised search heuristics such as evolutionary algorithms and swarm intelligence. He is an editorial board member of Evolutionary Computation and Natural Computing and received funding from the EU's Future and Emerging Technologies scheme (SAGE project). He has more than 70 refereed publications, including 8 best paper awards at GECCO and PPSN.

Andrew M. Sutton received the M.S. and Ph.D. degrees in computer science from Colorado State University, Fort Collins, CO, USA, in 2006 and 2011, respectively.

He has held Post-Doctoral Research fellowships with the University of Adelaide, Adelaide, SA, Australia, and the University of Jena, Jena, Germany. He is currently a Researcher with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany. His current research interest includes the theoretical analysis of randomised search heuristics.