How Crossover Helps in Pseudo-Boolean Optimization

Timo Kötzing
Max-Planck-Institut für Informatik
66123 Saarbrücken, Germany

Dirk Sudholt
CERCIA, University of Birmingham
Birmingham B15 2TT, UK

Madeleine Theile
Technische Universität Berlin
10623 Berlin, Germany

ABSTRACT

Understanding the impact of crossover on performance is a major problem in the theory of genetic algorithms (GAs). We present new insight on working principles of crossover by analyzing the performance of crossover-based GAs on the simple functions OneMax and Jump.

First, we assess the potential speedup by crossover when combined with a fitness-invariant bit shuffling operator that simulates a lineage of independent evolution on a function of unitation. Theoretical and empirical results show drastic speedups for both functions.

Second, we consider a simple GA without shuffling and investigate the interplay of mutation and crossover on Jump. If the crossover probability is small, subsequent mutations create sufficient diversity, even for very small populations. Contrarily, with high crossover probabilities crossover tends to lose diversity more quickly than mutation can create it. This has a drastic impact on the performance on Jump. We complement our theoretical findings by Monte Carlo simulations on the population diversity.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms

Algorithms, Design, Performance, Theory

Keywords

Crossover, recombination, pseudo-Boolean optimization, runtime analysis

1. INTRODUCTION

Crossover, also called recombination, is regarded an essential operator in genetic algorithms. Countless applications and empirical studies have shown that genetic algorithms with crossover are more effective than comparable algorithms using mutation only. While mutation typically makes only small changes to a genotype, crossover can combine genetic material found in different genotypes. One of the most natural settings for crossover is pseudo-Boolean optimization. Individuals are represented by an n-bit string and the fitness function assigns real values to each individual.

Schema theory was proposed to explain the successful use of crossover. A schema is a Boolean subspace where certain bits have fixed values and other bits can be set arbitrarily. Schema theory then deals with the growth of individuals belonging to schemas associated with high fitness values, in one generation of a GA. The building-block hypothesis states that using crossover GAs are able to combine good "building blocks", that is, schemata with few defining bits that contribute to a high fitness. However, there is neither a theoretical foundation nor convincing empirical evidence supporting this claim [24]. Another shortcoming of schema theory is that it is very hard to make predictions for more than one generation of a GA.

Runtime analysis has emerged as a different kind of theory. It is much less ambitious in terms of generality as one considers specific algorithms on specific problems or problem classes. But it allows to estimate the performance of genetic and evolutionary algorithms in a very precise way. Bounds on the running time are obtained until an evolutionary algorithm (EA) has found a desirable solution such as a global optimum. This time is called optimization time. The resulting bounds apply to problems of growing size and hence allow to assess the scalability of EAs.

Many such results have been obtained in the past 15 years, see, e. g., Droste, Jansen, and Wegener [6], Wegener [23], Oliveto, He, and Yao [18], or the recent book by Neumann and Witt [17]. The vast majority of these results focuses on EAs using mutation only, and studies are often limited to variants of the (1+1) EA using a population size of 1. One reason is that crossover is notoriously difficult to analyze. In order to analytically handle the effect of crossover, it is necessary to have a precise understanding of the dynamics within the population. The analysis of populations is much more difficult than the analysis of an (1+1) EA. Even on very simple problems like OneMax understanding the effect of crossover is still a tough challenge and an important open problem.

It is therefore not surprising that theoretical results on crossover after more than a decade of intensive research are still scarce. Speedups by crossover could be shown for coloring problems inspired by the Ising model [8, 21] and for

GECCO'11, July 12–16, 2011, Dublin, Ireland.

the all-pairs shortest path problem [3, 5, 4, 16, 11]. In terms of pseudo-Boolean optimization, most results were actually limited to artificial functions [13, 22, 19, 20] constructed such that a theoretical analysis was possible.

One notable exception is the first proof that crossover leads to a speedup by Jansen and Wegener [12]. They considered a simple test function $\textsc{Jump}_k$ and showed that uniform crossover can reduce a superpolynomial optimization time to a polynomial one. If $||x||_1$ denotes the number of ones in the bit string x, the function is defined as follows.

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

The function leads the population of a genetic algorithm to a plateau of search points with equal fitness. All search points on the plateau have $n-k$ ones. The optimum can be generated by a uniform crossover in case there is a pair of sufficiently diverse parents in the population. This operation is far more efficient than using mutation to jump to the optimum; such a mutation has probability at most n^{-k}.

As crossover heavily relies on diversity in the population, we first consider a setting for uniform crossover with a rather ideal diversity. This diversity is obtained by combining crossover with a specialized but *fitness-invariant* operator that is not found in typical GAs. The idea is to randomly shuffle all bits in the bit string before performing a uniform crossover. This does not affect the fitness on functions of unitation (i. e. functions depending on the number of ones only), but it simulates the potential diversity if we had run two independent strands of evolution on a unitation function. As such, it can create the diversity necessary to effectively perform crossover. Note that this crossover, called *shuffle crossover*, is not a geometric crossover as defined in [15]. The purpose is to assess the potential speedup of crossover in the presence of ideal diversity. We prove that this can lower the expected running time drastically—compared to the (1+1) EA without crossover—on $\textsc{OneMax}$ from $\Theta(n \log n)$ to $\Theta(\sqrt{n})$ function evaluations and on $\textsc{Jump}_k$ from $\Theta(n^k)$ to $\Theta(\sqrt{n} + 4^k)$. This illuminates the possible advantage crossover can have over mutation, depending on diversity.

For GAs without shuffling diversity is usually obtained by having large populations. The GA considered by Jansen and Wegener [12] needs a quite large population and a very low crossover probability to be effective on $\textsc{Jump}_k$. We show that large populations are not necessary to provide the required diversity if the crossover probability is small. This holds even up to population sizes as small as 2. On the other hand, large crossover probabilities make small populations collapse quickly, rendering crossover inefficient.

Thereby, we discuss an interesting effect. With standard, geometric crossovers EAs perform a convex search on the population, gradually decreasing the convex hull spanned by all members of the population until it eventually collapses to a single point [15]. Contrarily, mutation can create new points outside the present convex hull and thus randomly extends the convex hull. The interplay of these conflicting operators leads to an equilibrium state. The diversity in such an equilibrium state is essential for optimization. We use Monte Carlo simulations to gain deeper insights into this equilibrium and the respective diversity for various parameter settings on the $\textsc{Jump}$ plateau. This leads to novel insights on when crossover can create a global optimum.

2. ON THE POTENTIAL OF CROSSOVER

In contrast to mutation operators that typically only make small changes, crossover can yield a larger progress and a greater increase in fitness. This, however, only holds if the diversity in the population is large enough: if all individuals in the population are very similar, crossover will create a similar offspring. The analysis of diversity is a challenging problem in its own right [9]. We are interested in the *potential* of crossover. How large can the progress by crossover be if there is sufficient diversity in the population?

To answer this question we consider uniform crossover for a population where there is an "ideal" diversity. It is ideal in a sense that it models the situation where two individuals have been evolved independently to the same fitness value. On functions of unitation the fitness only depends on the number of bits set to 1. If mutation and/or uniform crossover is used (as opposed to 1-point or k-point crossover where linkage of bits comes into play), the position of 1-bits will be uniform if we look at one specific individual in the population.

We consider a shuffling operator that randomly permutes all bits in order to simulate a population that has evolved independently. For functions of unitation this operator is reasonable as it does not affect the fitness. It needs to be remarked, though, that this operator is clearly artificial and tailored towards functions of unitation. In particular, results obtained in this setting do not generalize under straightforward transformations of the search space such as exchanging bits values for selected bits. This in particular eludes lower bounds on the black-box complexity for such generalizations [7, 14]. Nevertheless, it allows us to observe the potential progress by crossover in a clear fashion.

The following algorithm called *shuffle GA* uses a population size 1 and the shuffling mechanism to create a second parent out of the current search point. This idea is similar to the gene invariant GA (GIGA) [1, 2].

Algorithm 1: Shuffle GA.

```
1 Initialize an individual x uniformly at random;
2 while true do
3     Let x' := x. Permute all bits of x' unif. at random;
4     Let y := uniform crossover(x, x');
5     if f(y) ≥ f(x) then x := y;
6 end
```

We abbreviate the two operations shuffling bits and performing a uniform crossover as *shuffle crossover*. Note that, for functions of unitation, it does not matter whether we shuffle one or both parents.

In the remainder of this section we determine tight bounds on the optimization time for the functions $\textsc{OneMax}$ and $\textsc{Jump}_k$. $\textsc{OneMax}$ represents a simple test case where all bits can be seen as small building blocks that have to be assembled. In contrast to the standard (1+1) EA [6] that needs $\Theta(n \log n)$ steps in expectation, uniform crossover can lead to a significant progress. If both parents have an equal number of 1-bits and their Hamming distance is k, then the gain of the $\textsc{OneMax}$-value for the offspring is governed by k independent Bernoulli trials. The following lemma shows that there is a good chance of having a surplus of $\Omega(\sqrt{k})$ ones.

LEMMA 1. *Let X be the sum of independent random variables $X_1, \dots, X_k \in \{0,1\}$ where $X_i = 1$ with probability $1/2$. Define the surplus of ones as $Y := \max\{0, X - k/2\}$. Then $\text{Prob}(Y \geq \pi/(4e) \cdot \sqrt{k}) \geq 1/4$ and $\text{E}(Y) \leq \sqrt{k} + O(1)$.*

PROOF. W. l. o. g. k is even. For every $0 \leq i \leq k$

$$\text{Prob}(X = i) \leq \text{Prob}(X = k/2) = \binom{k}{k/2} \cdot 2^{-k} = \frac{k! \cdot 2^{-k}}{((k/2)!)^2}.$$

From Stirling's approximation and a simple case analysis it follows for $k \in \mathbb{N}$ that $\sqrt{2\pi k} \cdot (k/e)^k \leq k! \leq e\sqrt{k} \cdot (k/e)^k$. Hence the above probability is at most

$$\frac{e\sqrt{k} \cdot (k/e)^k \cdot 2^{-k}}{(\sqrt{2\pi \cdot k/2})^2 \cdot (k/(2e))^k} = \frac{e}{\pi\sqrt{k}}.$$

As $\text{Prob}(X < k/2) = \text{Prob}(X > k/2) \leq 1/2$, we have that

$$\text{Prob}\left(X \leq k/2 + \frac{\pi}{4e} \cdot \sqrt{k} - 1\right) \leq \frac{1}{2} + \frac{\pi\sqrt{k}}{4e} \cdot \frac{e}{\pi\sqrt{k}} = \frac{3}{4}.$$

This proves the claimed probability bound for Y.

For the upper bound on $\text{E}(Y)$ we have

$$\text{E}(Y) = \sum_{i=0}^{k/2} \text{Prob}(Y \geq i) = \sum_{i=0}^{k/2} \text{Prob}(X \geq k/2 + i).$$

Clearly, $\text{Prob}(X \geq k) = 2^{-k}$ and $\text{E}(X) = k/2$. For $i < k/2$ we apply well known Chernoff bounds to get

$$\begin{aligned}\text{E}(Y) &= 2^{-k} + \sum_{i=0}^{k/2-1} \text{Prob}\left(X \geq \left(1 + \frac{2i}{k}\right) \cdot \frac{k}{2}\right) \\ &\leq 2^{-k} + \sum_{i=0}^{k/2-1} e^{-k/6 \cdot (2i/k)^2} = 2^{-k} + \sum_{i=0}^{k/2-1} e^{-2/3 \cdot i^2/k}.\end{aligned}$$

Estimating the first $\sqrt{k}$ summands of the $\sum$-term (w. l. o. g. assuming $\sqrt{k} \in \mathbb{N}$) by the trivial bound 1 and using $i^2 = (\sqrt{k} + (i - \sqrt{k}))^2 \geq k + (i - \sqrt{k})$ for $i \geq \sqrt{k}$, we arrive at

$$\begin{aligned}&\sqrt{k} + 2^{-k} + \sum_{i=\sqrt{k}}^{k/2-1} e^{-2/3 \cdot (i - \sqrt{k})} \\ &\leq \sqrt{k} + \sum_{i=0}^{\infty} e^{-2/3 \cdot i} = \sqrt{k} + \frac{1}{1 - e^{-2/3}}.\end{aligned}$$

□

The following theorem gives an upper bound on the optimization time of the shuffle GA on ONEMAX. As the shuffle GA cannot generate 1-bits when starting with 0^n, we have to exclude this case.

THEOREM 2. *Unless initialized with 0^n, the expected optimization time of the shuffle GA on* ONEMAX *is $O(\sqrt{n})$.*

PROOF. We estimate the increase of the offspring's ONEMAX-value using the surplus argument of Lemma 1. As the crossover operator can only yield a surplus on bits where both parents differ we first derive a lower bound on the number of these bits.

Let $k < n$ be the number of zeros in the current bit string x. First consider the case $k \leq n/2$. Consider the bit string x' obtained by shuffling x. Whenever there is a bit where both x and x' are 0, we speak of a collision. For a fixed 0-bit in x we have that the expected number of collisions

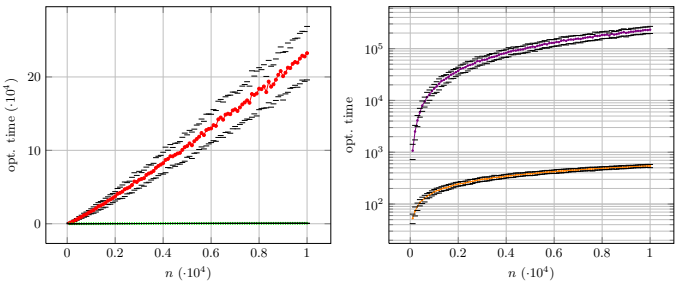

Figure 1: Average optimization times and standard deviation (–) for the (1+1) EA (—●—) and the shuffle GA (—+—) on OneMax on linear (left) and logarithmic (right) scales

of the bit is k/n. By linearity of expectation, the total expected number of collisions is therefore $k \cdot k/n = k^2/n$. Using Markov's inequality, with probability at least $1/3$ we have at most $3/2 \cdot k^2/n$ collisions. This implies that the number of 0-bits in x where x' has value 1 is at least $k - 2k^2/n \geq k/4$ with probability at least $1/3$. The case $k \geq n/2$ follows by symmetric arguments, exchanging the roles of zeros and ones.

Along with Lemma 1, the shuffle GA gains at least $\pi/(8e) \cdot \sqrt{k}$ ones with probability at least $1/12$. Other steps cannot decrease the number of ones. If, for the current number of zeros k, it holds that $n/(2^{i+1}) < k \leq n/2^i$ for some $i \in \mathbb{N}$, the expected time until this number has decreased to or below $n/2^{i+1}$ is at most

$$12 \cdot \left\lceil \frac{n}{2^i} \cdot \frac{8e}{\pi\sqrt{n/2^{i+1}}} \right\rceil \leq 12 + \frac{96e}{\sqrt{2}\pi} \cdot \frac{\sqrt{n}}{2^{i/2}}.$$

As we start with at most $n/2^0$ zeros and finish when there are at most $n/2^{(\log n)+1} < 1$ zeros left, summing up all expected time bounds, the total expected optimization time is

$$\begin{aligned}&\sum_{i=0}^{(\log n)+1} \left(12 + \frac{96e}{\sqrt{2}\pi} \cdot \frac{\sqrt{n}}{2^{i/2}}\right) \\ &\leq O(\log n) + \frac{96e}{\sqrt{2}\pi} \cdot \sqrt{n} \cdot \sum_{i=0}^{\infty} 2^{-i/2} = O(\sqrt{n}).\end{aligned}$$

□

The speedup compared to the time $\Theta(n \log n)$ of the (1+1) EA is dramatic. To see that this is not just a theoretical artifact, we consider experiments for realistic problem dimensions. Figure 1 shows average optimization times for $n \in \{100, 200, 300, \dots, 10.000\}$ with a hundred runs for each n. The plot on the left shows the huge difference of the runtime between a standard (1+1) EA on ONEMAX and our shuffle GA. The plot on the right uses a logarithmic scale. Note that the variance for the shuffle GA is smaller than for the (1+1) EA, i. e. the runtime of the shuffle GA is much more concentrated.

The reason for the overwhelming success of the shuffle GA is that shuffling bits of an individual with i zeros creates another individual uniformly distributed on the Hamming ball around the optimum 1^n. Recombining two parents that

are "on opposite sides" of the global optimum can yield a significant progress towards the optimum.

In order to avoid misinterpretations, we stress that Theorem 2 does not contradict higher lower bounds on the black-box complexity of a class of generalized OneMax functions [7, 14]. We are only considering one specific function and the black-box complexity of a single function is trivially 1 [7].

The upper bound for OneMax is asymptotically tight. In fact, the following result holds for arbitrary functions that only have one global optimum. This presents a limit to the potential speedup by crossover.

Theorem 3. *The expected optimization time of the shuffle GA on any function with a unique optimum is* $\Omega(\sqrt{n})$.

Proof. Without loss of generality, we suppose the optimum to be the all-1 string. We use drift on the number of 0-bits in the individual. For all t, we let X_t be the random variable of this number after t iterations. It is easy to see that we can bound the drift of $(X_t)_{t\in\mathbb{N}}$ from above by supposing there is no negative drift, and that each search point with less 0-bits is accepted.

The number of bits differing in both parents when performing a uniform crossover is trivially bounded by n. By Lemma 1 this implies that the expected decrease of the number of zeros is always bounded by $\mathrm{E}(X_t - X_{t+1} \mid X_t) = O(\sqrt{n})$.

The lower bound now follows from well known additive drift results (Lemma 2 in [10]), saying that if always $\mathrm{E}(X_t - X_{t+1} \mid X_t) \leq \delta$ holds for some $\delta > 0$ then the expected time until $X_t = 0$ is at most $\mathrm{E}(X_0)/\delta$. Here the expected number of zeros in the initial search point is $\mathrm{E}(X_0) = n/2$. Setting $\delta := O(\sqrt{n})$, we get a lower bound of $n/(2\delta) = \Omega(\sqrt{n})$. □

For $\textsc{Jump}_k$ shuffling a search point on the plateau creates another search point on the plateau. The resulting diversity is sufficient to show the following result.

Theorem 4. *Unless initialized with* 0^n*, the expected optimization time of the shuffle GA on* $\textsc{Jump}_k$ *with* $k \leq \sqrt{n/2}$ *is* $\Theta(\sqrt{n} + 4^k)$.

This is a significant speedup, compared to the standard (1+1) EA that needs expected time $\Theta(n \log n + n^k)$ [12]. For $k := \log n$, say, this is a difference between polynomial and superpolynomial running times.

Proof of Theorem 4. Following the analysis of the proof of Theorem 2, the individual will have reached the plateau of fitness k after $O(\sqrt{n})$ generations. The probability that a single application of the shuffle crossover now produces the optimum is the probability of choosing disjoint sets of bit positions in the two random permutations of the individual for the 0s, and then choosing a 1 at all these positions in the uniform crossover. As there are $2k$ positions where the selected parents differ, the probability for uniform crossover setting all bits to 1 is $2^{-2k} = 4^{-k}$. The probability of obtaining the right parents can be bounded as follows. Fixing the k positions for 0-bits in the first individual, consider all k positions for the second individual to be assigned sequentially. All zeros of the second individual must not be in at most $2k$ positions that are either fixed or already used. Thus, we get a lower bound on the probability of

$$\left(1 - \frac{2k}{n}\right)^k \cdot 4^{-k} \geq \left(1 - \frac{k^2}{n}\right) \cdot 4^{-k}.$$

The upper bound follows from $k^2 \leq \frac{n}{2}$.

It is easy to see that with overwhelming probability no search point with more than $n - k$ ones and less than n ones is accepted after initialization. The lower bound then follows from the general lower bound $\Omega(\sqrt{n})$ and the fact that uniform crossover of two individuals with at least k zeros each is successful with probability at most 4^{-k}. □

We get the following corollary to the theorems above.

Corollary 5. *Unless initialized with* 0^n*, the expected optimization time of the shuffle GA on* OneMax *and* $\textsc{Jump}_k$ *(*$k \leq \frac{\log n}{4}$*) is* $\Theta(\sqrt{n})$.

3. POPULATION SIZE VS. CROSSOVER PROBABILITY

The consideration of the shuffle GA has shown that diversity is essential for achieving speedups with crossover. We now turn from an "ideal" setting to a more realistic GA. Without the shuffling operator one might think that a large population is necessary to provide diversity. However, if the crossover probability is small, mutation can create diversity in subsequent steps. We make this precise for the function $\textsc{Jump}_k$ by showing that small populations, even of size as small as $\mu = 2$, suffice to optimize $\textsc{Jump}_k$ efficiently.

Jansen and Wegener [12] considered a steady-state genetic algorithm as given in Algorithm 2. In their analysis on the function $\textsc{Jump}_k$ they disallowed mutation to create replicates of the parent. The reason is that otherwise replicates would quickly take over the population and so diversity is lost. So, when mutation does not flip any bit, the offspring is disregarded. Note that this does not apply to mutation following crossover.

Algorithm 2: (μ+1) GA with avoidance of replications by mutation and with/without mutation after crossover.

```
1  Initialize population P of size μ ∈ ℕ uniformly at
   random, choose p_c ∈ (0,1);
2  while true do
3      Choose q ∈ [0,1] uniformly at random;
4      if q ≤ p_c then
5          Choose y_1, y_2 uniformly at random from P;
6          Obtain y' by a uniform crossover of y_1 and y_2;
7          if use mutation after crossover then
8              Flip each bit in y' independently with
               probability 1/n;
9          end
10     end
11     else
12         Choose y uniformly at random from P;
13         Create y' by flipping each bit in y independently
           with probability 1/n;
14         if y = y' then
15             Continue at line 3 to avoid replicates;
16         end
17     end
18     Choose z ∈ P with minimal fitness in P u. a. r.;
19     if f(y') ≥ f(z) then
20         Let P := P \ {z} ∪ {y'};
21     end
22 end
```

Their result for $\textsc{Jump}_k$ reads as follows (they also prove a tail bound, but we only review the result on the expected optimization time).

THEOREM 6 (JANSEN AND WEGENER [12]). *Let $k = O(\log n)$. Consider the $(\mu+1)$ GA with crossover probability $p_c \leq 1/(ckn)$ (where c is a large enough constant), and population size μ where $\mu \geq k \log^2 n$ and $\mu = n^{O(1)}$ on $\textsc{Jump}_k$. The expected optimization time is $O(\mu n(k^2 + \log(\mu n)) + 4^k/p_c)$.*

The behavior of the $(\mu+1)$ GA on the function $\textsc{Jump}_k$ is of particular interest for us, as $\textsc{Jump}_k$ cannot be optimized efficiently by mutation only, while a crossover has to rely on a sufficient diversity in the population: the population easily reaches the locally optimal plateau of fitness n, but will then need to create what we call *complementary pairs*—pairs of individuals which do not have a 0 at a common bit position, and which can thus be recombined to reach the optimum.

In this section we give conditions on the parameters of the $(\mu+1)$ GA which lead to efficient optimization of $\textsc{Jump}_k$, as well as conditions which lead to expected superpolynomial optimization times.

First, we prove that the $(\mu+1)$ GA also works with very small populations and not too big crossover probability. In particular, the GA is effective using the smallest possible population size: $\mu = 2$. In addition, larger crossover probabilities can be used, compared to the result by Jansen and Wegener [12]. Instead of requiring $p_c \leq 1/(ckn)$, the following proof works for $p_c \leq k/n$.

THEOREM 7. *The expected optimization time of the $(\mu+1)$ GA with or without mutation after crossover, $\mu \geq 2$, $\mu \leq n^{O(1)}$, and $p_c \leq \frac{k}{n}$ on $\textsc{Jump}_k$ with $k = o(\sqrt{n})$ is $O\big(\mu n \log n + e^{6k} \cdot \mu^{k+2} \cdot n\big)$.*

Theorem 7 gives the smallest upper bound for $\mu = 2$. In this case for every $\varepsilon > 0$, if $k = c \log n$ for a sufficiently small constant c depending on ε, this gives an expectation of $O(n^{1+\varepsilon})$, while mutation-based EAs without crossover still need superpolynomial time $\Omega(n^{c \log n})$.

If $\mu = 2$ and $k = c \log \log n$ for a sufficiently small c, we get $O(n \log n)$, while mutation-based EAs without crossover need time $\Omega(n^{c \log \log n})$, which is still superpolynomial.

PROOF OF THEOREM 7. It is easy to show that when $\mu = n^{O(1)}$ then in expected time $O(\mu n \log n)$ the whole population contains only the optimum 1^n or search points on the plateau (cf. the proof of Theorem 5 in [12]). Note that this property will be preserved forever as all other search points have worse fitness (apart from the optimum).

A population is called *perfect* if each member has $n - k$ ones and there is a complementary pair in the population. Note that uniform crossover can only create the global optimum 1^n in a perfect population.

We describe a sequence of lucky events that results in a perfect population. We call this sequence a *trial*. The length of such a sequence is bounded. A trial is called *successful* if it leads to a crossover operation on a perfect population. In a perfect population there are two parents having k zeros at different positions. This means that their Hamming distance is $2k$ and all these bits have to be set to 1 in a uniform crossover. The probability that a successful trial creates a global optimum is then at least $1/\binom{\mu}{2} \cdot 2^{-2k}$ if no mutation is used after crossover and $1/\binom{\mu}{2} \cdot 2^{-2k} \left(1 - \frac{1}{n}\right)^n$ otherwise, as the latter factor describes the probability that mutation does not flip any bit.

The expected optimization time then follows from the probability of a trial being successful, the length of a trial, and the above probability of creating an optimum at the end of a successful trial.

Consider a population where all members have $n-k$ ones. For the next at most $n+n/k$ generations we define the following events. The first events concern the first n generations, or a shorter time span if a perfect population or a global optimum is found earlier. The last event is based on the following n/k generations. This means that a trial contains at most $n + n/k$ generations. We pessimistically ignore the possibility that a perfect population or a global optimum might be reached prematurely.

E1: Within n generations no crossover is performed. This event has probability at least $(1 - p_c)^n \geq (1 - k/n)^n \geq e^{-k}(1 - o(1))$.

E2: Conditional on E1, within n generations at least k mutations are performed such that exactly two bits are flipped and the offspring has $n - k$ 1-bits. Such a mutation has probability $p := k/n \cdot (n-k)/n \cdot (1-1/n)^{n-2}$ as exactly one 1-bit and one 0-bit have to be flipped. Assuming n is large enough such that $(n-k)/n \cdot (1-1/n)^{n-2} \geq 1/3$, we have $k/(3n) \leq p \leq k/n$. The probability of the described event is at least

$$\begin{aligned}\binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} &\geq \frac{n^k}{k^k} \cdot \frac{k^k}{n^k} \cdot 3^{-k} \cdot (1-p)^{n-k} \\ &\geq 3^{-k} \cdot \left(1 - \frac{k}{n}\right)^{\left(\frac{n}{k}-1\right)\cdot k} \geq (3e)^{-k}.\end{aligned}$$

E3: Conditional on E1, within n generations it never happens that mutation creates an offspring with Hamming distance larger than 2 to its parent such that the offspring has $n - k$ ones. The probability for a single such mutation is at most $\binom{k}{2} \cdot 1/n^2 \leq k^2/n^2$ as two 0-bits need to be flipped. The event E3 thus has probability at least

$$\left(1 - \frac{k^2}{n^2}\right)^n \geq 1 - O\left(\frac{k^2}{n}\right) \geq 1 - o(1)$$

as $k = o(\sqrt{n})$. Note that E2 and E3 are not independent, but $\text{Prob}(\text{E2} \wedge \text{E3}) \geq \text{Prob}(\text{E2}) \cdot \text{Prob}(\text{E3})$.

E4: Assume E1, E2, and E3 happen, which implies that we have at most k generations where the population changes, due to the avoidance of replications. E4 is the event that selection in such a generation always chooses appropriate individuals in the following sense. A parent x is selected which is part of a pair x, y of individuals that have a maximal Hamming distance in the population: $x, y = \arg\max\{H(x', y') \mid x', y' \in \mathcal{P}\}$. Also the individual to be removed is selected in such a way that the above maximum pairwise Hamming distance in the population is not decreased.

The probability of always choosing a parent as described is at least $(2/\mu)^k$ as there is at least one pair of individuals defining the maximal Hamming distance. For the same reason the probability of always choosing the individual to be removed as described is at least $((\mu-1)/(\mu+1))^k \geq (1/3)^k$. Together, the probability for E4 is at least $(2/3)^k \mu^{-k}$.

E5: Conditional on E1–E4, the maximum Hamming distance of pairs in the current population never decreases.

Due to E4 in every accepted mutation a parent x is selected such that it forms a pair of maximal Hamming dis-

tance with some other member y of the population. Let these individuals have Hamming distance $2i$. We estimate the probability that the result of mutating x increases the Hamming distance to y in an accepted 2-bit mutation. This distance increases from its current value $2i$ if mutation flips a 0-bit where x and y agree as well as a 1-bit where x and y agree. There are i bits where x is 0 and y is 1 and i bits where this is the other way round. Conditional on an accepted 2-bit mutation, as all 0-bits and 1-bits are chosen with the same probability, respectively, the probability of choosing bits where x and y agree is $(k-i)/k \cdot (n-k-i)/(n-k)$. The probability of making a sequence of these mutations is thus at least

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

E6: Conditional on E1–E5, consider the situation after a perfect population has been reached. Note that from now on we work without conditioning on the events E1–E5. So, we take a fresh look and define E6 as the event that in the following n/k generations at least one crossover happens and no mutation leads to an accepted offspring. Note that a necessary event for an accepted mutation is that at least one of k 0-bits flips. The probability for the event E6 is at least

$$1-(1-p_c)^{n/k}\cdot\left(1-\frac{k}{n}\right)^{k/n} = \Omega(1).$$

If E1–E6 happen, the trial is successful. The probability of E1–E6 happening is

$$\begin{aligned}&e^{-k}\cdot(3e)^{-k}\cdot(2/3)^k\mu^{-k}\cdot e^{-k}\cdot(1-o(1))^3\cdot\Omega(1)\\ &=\Omega\left(2^k\cdot 3^{-2k}\cdot e^{-3k}\cdot\mu^{-k}\right).\end{aligned}$$

The length of a trial is at most $n+n/k = O(n)$ by definition. Hence, starting from a population with $n-k$ ones in every individual, the expected time until the global optimum is created is

$$O\left(2^{-k}\cdot 3^{2k}\cdot e^{3k}\cdot\mu^k\cdot n\cdot\binom{\mu}{2}\cdot 2^{2k}\right) = O\left(e^{6k}\cdot\mu^{k+2}\cdot n\right). \quad \square$$

The (μ+1) GA is effective for small populations if the crossover probability is fairly small. In this case the explorative effects of mutation are larger than the effect of crossover subsequently compressing the convex hull of the population.

We complement this finding by an opposite result for large crossover probabilities. If the crossover probability is an arbitrary constant, the population is compressed to a single point more quickly than mutation can create diversity. This holds for up to logarithmic population sizes.

THEOREM 8. *Consider the (μ+1) GA* without *mutation after crossover on* $\textsc{Jump}_k$ *with* $k = \log n$. *For every* $p_c = \Omega(1)$ *there is a constant* $c > 0$, *depending on* p_c, *such that if* $\mu \leq c \log n$ *then the expected optimization time of the (μ+1) GA is superpolynomial.*

PROOF. It is easy to see that after the first $O(\mu n \log n) = o(n^2)$ generations of the GA, with high probability, the optimum is not created (since neither mutation nor crossover can bridge the gap of the $\textsc{Jump}_k$ function in this time with sufficient probability), and the whole population is on the plateau.

Suppose now we have the whole population on the plateau. We show that, with high probability, an arbitrary population collapses to copies of a single individual within a short number of steps. More precisely, for every population on the plateau with probability at least $1/\sqrt{n}$ the whole population will consist of copies of a single individual after μ generations. A sufficient condition for this to happen is that in each generation two equal parents are selected, crossover is performed, and a proper individual is chosen for removal. If there are i equal individuals in the population, increasing this number to $i+1$ has probability at least

$$\frac{i^2}{\mu^2}\cdot p_c\cdot\frac{\mu-i}{\mu+1}.$$

The probability of this always happening is thus at least

$$\begin{aligned}\prod_{i=1}^{\mu-1}\left(\frac{i^2}{\mu^2}\cdot p_c\cdot\frac{\mu-i}{\mu+1}\right) &= p_c^{\mu-1}\cdot\left(\frac{1}{\mu^2(\mu+1)}\right)^{\mu-1}\cdot((\mu-1)!)^3\\ &\geq p_c^{\mu-1}\cdot e^{-3(\mu-1)}\left(\frac{(\mu-1)^3}{\mu^2(\mu+1)}\right)^{\mu-1}\end{aligned}$$

and as the bracketed term is $\Omega(1)$ this probability is at least $1/\sqrt{n}$ if the constant c from the preconditions is sufficiently small. The probability that this collapse never happens in $t := \sqrt{n}\mu k \ln n$ generations is at most $(1-1/\sqrt{n})^{\sqrt{n}k\ln n} \leq n^{-k}$.

Assuming the population has collapsed, we show that in t steps with high probability no complementary pair is created. Without mutation after crossover, crossover then cannot create the global optimum. Complementary pairs can only be created if, starting with a collapsed population, over time in total k 0-bits have been flipped. The expected number of flipping 0-bits in t mutations is tk/n. By Chernoff bounds, the probability that at least k bits are flipped in total in this time span is

$$\left(\frac{e^{n/t-1}}{(n/t)^{n/t}}\right)^{kt/n} \leq \left(\frac{e^{n/t}}{(n/t)^{n/t}}\right)^{kt/n} = \left(\frac{et}{n}\right)^k.$$

By definition of t and recalling $k = \log n$, this probability is superpolynomially small. Hence, with a superpolynomially small error probability the population collapses again before a complementary pair is created.

The probability of creating the global optimum by mutation is at most n^{-k} and hence superpolynomially small as well. As one of the mentioned unlikely events must occur in order to create the global optimum, this proves a superpolynomial lower bound. $\square$

4. EXPERIMENTAL RESULTS FOR POPULATION DIVERSITY

Finally, we add experimental results about the behavior of the (μ+1) GA on the plateau of $\textsc{Jump}_k$. We are particularly interested in the population diversity given by the distribution of Hamming distances between individuals. The most important case is the number of complementary pairs;

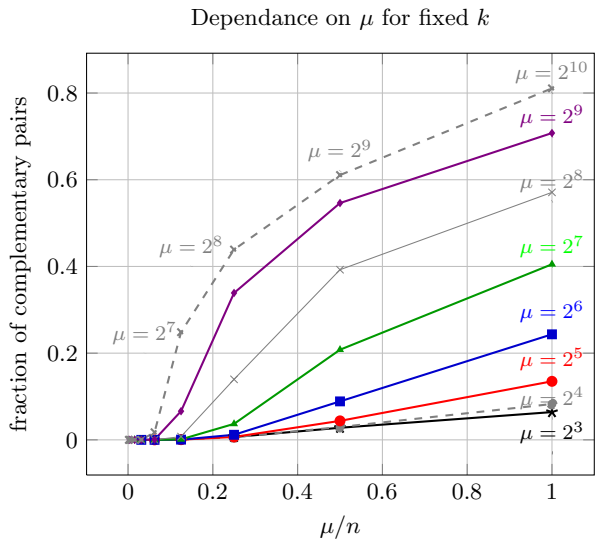

Figure 2: Fraction of complementary pairs vs. normalized population sizes μ/n, for the $(\mu+1)$ GA on Jump_k with $k = 3$ (—★—), $k = 4$ (- -●- -), $k = 5$ (—●—), $k = 6$ (—■—), $k = 7$ (—▲—), $k = 8$ (—×—), $k = 9$ (—♦—), and $k = 10$ (- -*- -).

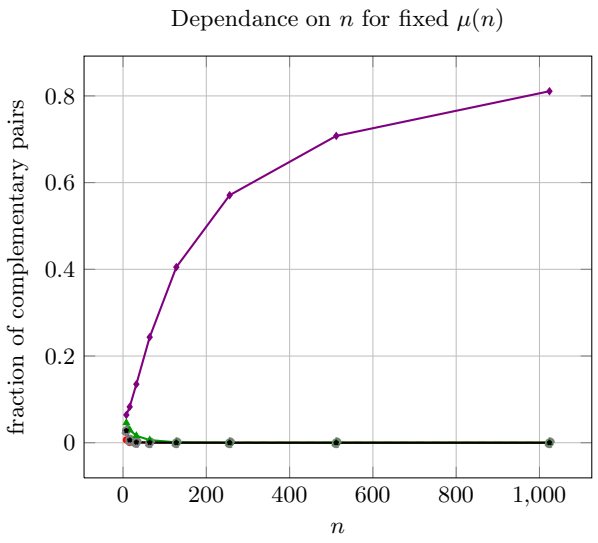

Figure 3: Fraction of complementary pairs vs. problem dimensions, for the $(\mu+1)$ GA on Jump_k with population size functions $\mu(n) = 2$ (—●—), $\mu(n) = 4$ (- -●- -), $\mu(n) = \log(n)$ (—▲—), $\mu = \sqrt{n}$ (—•—), and $\mu = n$ (—♦—).

the existence of many such pairs guarantees that uniform crossover is effective as 1^n can only be created if complementary parents are selected.

We set the crossover probability to $p_c = 0.5$ and allowed mutations after a crossover step. We conducted experiments for various parameters n and with $k = \log n$, as well as different population sizes μ. All results are averaged over 100 runs for each experiment.

We measured the number of complementary pairs in each generation of the population on the plateau. In order to estimate the diversity at an equilibrium state, we removed the optimum of the function Jump_k and stopped each run after 1 million generations. Even for $\mu = 1024$ and $n = 1024$, the population was gathered on the plateau mostly before generation $\approx$ 150 000 such that the remaining $\approx$ 850 000 generations the population could average out to the equilibrium.

The analysis of the $(\mu+1)$ GA by Jansen and Wegener [12] only works under limited conditions of a large population size and very small crossover probability in order to maintain diversity. One might get the impression that a large crossover probability is harmful for the population diversity regardless of their size.

Figure 2 shows for $k \in \{5, 6, \ldots, 10\}$ how the ratio μ/n of populations size and $n := 2^k$ influences the number of complementary pairs in the population. Figure 3 shows the dependency of the percentage of complementary pairs, for different population size functions $\mu(n) \in \{2, 4, \log(n), \sqrt{n}, n\}$. For both plots the different values of μ on the x axis are linearly interpolated.

Our experiments do not confirm the mentioned impression. Having used a rather large crossover probability, we find that the larger the population, the more complementary pairs are available. Figure 2 shows that, for a reasonably large population size, a sufficiently large fraction of pairs are complementary pairs. However, Figure 3 shows that in order to have a constant fraction or a fraction converging to

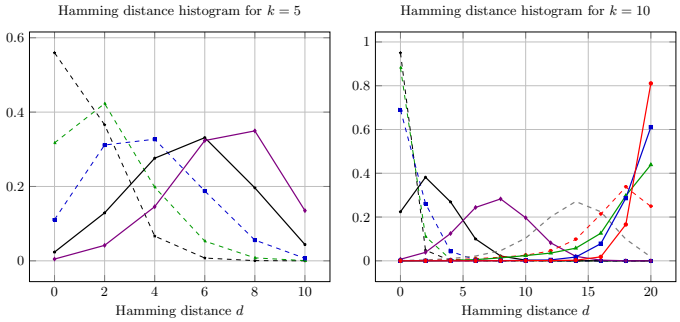

Figure 4: Histograms of pairwise Hamming distances in the population, averaged over time, for the $(\mu+1)$ GA on Jump_k with $k \in \{5, 10\}$, $n = 2^k$, and $\mu = 2$ (- -•- -), $\mu = 4$ (- -▲- -), $\mu = 8$ (- -■- -), $\mu = 16$ (—•—), $\mu = 32$ (—♦—), $\mu = 64$ (- - -), $\mu = 128$ (- -●- -), $\mu = 256$ (—▲—), $\mu = 512$ (—■—), $\mu = 1024$ (—●—).

1 the population size should depend on $n = 2^k$ in a linear fashion, i. e. $\mu = \Theta(n)$. For the tested population sizes $o(n)$ we do not see a constant fraction of the population being complementary pairs. On the other hand a population size growing linearly with n contains an overwhelming number of complementary pairs.

In order to gain further insights into the existence of complementary pairs, for $k \in \{3, \ldots, 10\}$ we computed a histogram of pairwise Hamming distances in the population, averaged over time (again with $n = 2^k$ and for various μ). For each generation on the plateau the pairwise Hamming distances $d \in \{0, \ldots, 2k\}$ (only even d can occur) were counted and the results were averaged over all generations on the plateau. Figure 4 shows two selected histograms for $k = 5$ and $k = 10$; the other histograms looked similar. One can see that the curves gradually shift from leaning towards small distances to Bell curves and leaning towards large distances as the population size increases. Note that

the highest value gives the average number of complementary pairs.

This also supports the claim that a large population size is needed in the presence of a large crossover probability. As can be seen here, the diversity for large population sizes is indeed significant, that is for $k = 10$ and $\mu \geq 256$ we hardly observe any pairs of individuals having a Hamming distance less than $2k$. But we also observe the effect of a collapsing population if the population size is not large enough. This supports the claim of Theorem 8 even in the presence of mutation after crossover, compare the lines for $k = 10$ and $\mu \leq 64$. Here, mutation is not able to create or maintain the needed diversity opposing the effect of crossover to reduce diversity when the population size is not large enough.

5. CONCLUSIONS

Crossover is a very useful operator if the population is sufficiently diverse. Our considerations of the shuffle GA have revealed the potential of uniform crossover in populations with good diversity. The bit shuffling operator simulates individuals that have evolved independently on a function of unitation. In this setting crossover leads to surprising and drastic speedups from $\Theta(n \log n)$ function evaluations to only $\Theta(\sqrt{n})$ for OneMax. For Jump_k the difference is between $\Theta(n^k)$ and $\Theta(\sqrt{n} + 4^k)$.

In a more realistic setting of the (μ+1) GA we have shown that also small populations can develop a sufficient diversity for Jump_k if the crossover probability is small enough. If it is set too large, crossover makes the population collapse to a single individual, leading to superpolynomial running times on Jump_k. Together with our Monte Carlo simulations we have provided a deepened insight into the opposing effects that mutation and crossover have on the convex hull and thus the diversity of the population.

Acknowledgments

Timo Kötzing was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant NE 1182/5-1. Dirk Sudholt was supported by EPSRC grant EP/D052785/1. The authors thank Thomas Sauerwald and Tobias Friedrich for their valuable comments and the organizers of the Theory of Evolutionary Algorithms seminar at Schloss Dagstuhl, where this research was initiated.

6. REFERENCES

[1] J. Culberson. Genetic invariance: A new paradigm for genetic algorithm design. Technical report, 1992.

[2] M. Dietzfelbinger, B. Naudts, C. Van Hoyweghen, and I. Wegener. The analysis of a recombinative hill-climber on H-IFF. *IEEE Transactions on Evolutionary Computation*, 7(5):417–423, 2003.

[3] B. Doerr, E. Happ, and C. Klein. Crossover can provably be useful in evolutionary computation. In *Proc. of GECCO '08*, pages 539–546. ACM, 2008.

[4] B. Doerr, D. Johannsen, T. Kötzing, F. Neumann, and M. Theile. More effective crossover operators for the all-pairs shortest path problem. In *Proc. of PPSN '10*, pages 184–193. Springer, 2010.

[5] B. Doerr and M. Theile. Improved analysis methods for crossover-based algorithms. In *Proc. of GECCO '09*, pages 247–254. ACM, 2009.

[6] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276:51–81, 2002.

[7] S. Droste, T. Jansen, and I. Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of Computing Systems*, 39(4):525–544, 2006.

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

[9] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt. Analysis of diversity-preserving mechanisms for global exploration. *Evolutionary Computation*, 17(4):455–476, 2009.

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

[11] C. Horoba and D. Sudholt. Running time analysis of ACO systems for shortest path problems. In *Proc. of SLS '09*, pages 76–91. Springer, 2009.

[12] T. Jansen and I. Wegener. The analysis of evolutionary algorithms - a proof that crossover really can help. *Algorithmica*, 34(1):47–66, 2002.

[13] T. Jansen and I. Wegener. Real royal road functions—where crossover provably is essential. *Discrete Applied Mathematics*, 149:111–125, 2005.

[14] P. K. Lehre and C. Witt. Black box search by unbiased variation. In *Proc. of GECCO '10*, pages 1441–1448, 2010.

[15] A. Moraglio. Convex evolutionary search. In *Proc. of FOGA '11*, pages 151–162. ACM, 2011.

[16] F. Neumann and M. Theile. How crossover speeds up evolutionary algorithms for the multi-criteria all-pairs-shortest-path problem. In *Proc. of PPSN '10*, pages 667–676. Springer, 2010.

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

[18] P. S. Oliveto, J. He, and X. Yao. Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results. *Int'l Journal of Automation and Computing*, 4(3):281–293, 2007.

[19] J. N. Richter, A. Wright, and J. Paxton. Ignoble trails - where crossover is provably harmful. In *Proc. of PPSN '08*, pages 92–101. Springer, 2008.

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

[21] D. Sudholt. Crossover is provably essential for the Ising model on trees. In *Proc. of GECCO '05*, pages 1161–1167. ACM, 2005.

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

[23] I. Wegener. Methods for the analysis of evolutionary algorithms on pseudo-Boolean functions. In R. Sarker, X. Yao, and M. Mohammadian, editors, *Evolutionary Optimization*, pages 349–369. Kluwer, 2002.

[24] A. H. Wright, M. D. Vose, and J. E. Rowe. Implicit parallelism. In *Proc. of GECCO '03*, pages 1505–1517. Springer, 2003.