Crossover for Cardinality Constrained Optimization

TOBIAS FRIEDRICH, TIMO KÖTZING, AISHWARYA RADHAKRISHNAN, and LEON SCHILLER, Hasso Plattner Institute, University of Potsdam, Germany
MARTIN SCHIRNECK, Faculty of Computer Science, University of Vienna, Austria
GEORG TENNIGKEIT and SIMON WIETHEGER, Hasso Plattner Institute, University of Potsdam, Germany

To understand better how and why crossover can benefit constrained optimization, we consider pseudo-Boolean functions with an upper bound B on the number of 1-bits allowed in the length-n bit string (i.e., a cardinality constraint). We investigate the natural translation of the OneMax test function to this setting, a linear function where B bits have a weight of $1 + 1/n$ and the remaining bits have a weight of 1. Friedrich et al. [TCS 2020] gave a bound of $\Theta(n^2)$ for the expected running time of the (1+1) EA on this function.

Part of the difficulty when optimizing this problem lies in having to improve individuals meeting the cardinality constraint by flipping a 1 and a 0 simultaneously. The experimental literature proposes balanced operators, preserving the number of 1-bits, as a remedy. We show that a balanced mutation operator optimizes the problem in $O(n \log n)$ if $n - B = O(1)$. However, if $n - B = \Theta(n)$, we show a bound of $\Omega(n^2)$, just as for classic bit mutation. Crossover together with a simple island model gives running times of $O(n^2/\log n)$ (uniform crossover) and $O(n\sqrt{n})$ (3-ary majority vote crossover). For balanced uniform crossover with Hamming-distance maximization for diversity, we show a bound of $O(n \log n)$.

As an additional contribution, we present an extensive analysis of different balanced crossover operators from the literature.

CCS Concepts: • **Theory of computation → Theory of randomized search heuristics**; *Optimization with randomized search heuristics*; *Evolutionary algorithms*;

Additional Key Words and Phrases: Balanced crossover, balanced mutation, constraint optimization, run time analysis.

ACM Reference format:
Tobias Friedrich, Timo Kötzing, Aishwarya Radhakrishnan, Leon Schiller, Martin Schirneck, Georg Tennigkeit, and Simon Wietheger. 2023. Crossover for Cardinality Constrained Optimization. *ACM Trans. Evol. Learn.* 3, 2, Article 5 (June 2023), 32 pages.
https://doi.org/10.1145/3603629

An extended abstract of this work was presented at the 2022 Genetic and Evolutionary Computation Conference (GECCO) [Friedrich et al. 2022]. The present work includes all proofs. It contains a new lower bound, showing a tight expected running time of $\Theta(n^2/\log(n))$ for the (2+1) island model with balanced uniform crossover if the constraint satisfies $n - B = \Theta(n)$. The upper bound of $O(n \log(n))$ for the (2+1) SWAP-GA with balanced uniform crossover and Hamming-distance maximization was generalized to all possible constraints B.

Authors' addresses: T. Friedrich, T. Kötzing, A. Radhakrishnan, L. Schiller, G. Tennigkeit, and S. Wietheger, Hasso Plattner Institute, University of Potsdam, Prof.-Dr.-Helmert-Straße 2-3, 14482, Potsdam, Germany; emails: {tobias.friedrich, timo.koetzing, aishwarya.radhakrishnan}@hpi.de, {leon.schiller, georg.tennigkeit, simon.wietheger}@student.hpi.de; M. Schirneck, Faculty of Computer Science, University of Vienna, Universitätsring 1, 1010, Vienna, Austria; email: martin.schirneck@hpi.de.

2688-3007/2023/06-ART5 $15.00
https://doi.org/10.1145/3603629

1 INTRODUCTION

The analysis of crossover, asking how, why, and when it helps optimization, is one of the most important topics for understanding evolutionary computation. In practice, hardly any application in the field proceeds without crossover. It is known to aid the search, but its theoretical background still remains somewhat in the dark.

Focusing on the optimization of pseudo-Boolean functions (that is, $f : \{0,1\}^n \to \mathbb{R}$), most theory literature on the topic has used the Jump function as a setting to understand crossover, showing that crossover can jump a fitness valley effectively where mutation fails. This line of work was started by the seminal [Jansen and Wegener 2002], showing that crossover can find the optimum of Jump in time $O(\mu n \log n)$, if it is used only very rarely (with probability of about $1/n$) and with sufficient population size of $\Omega(\log^2(n))$; this is in contrast to the time of $O(n^k)$ taken by mutation-based algorithms. This work was extended by Kötzing et al. [2011] to higher parameter ranges. The low crossover probability is crucial for the analysis: Fast optimization of Jump relies on generating very different individuals to be used as parents by crossover; mutation can generate diversity, while crossover is pessimistically assumed to reduce diversity.

Typical crossover probabilities are much higher than the required $1/n$; thus, further work focused on closing this gap. For the case of diversity mechanisms, Dang et al. [2016] shows efficient optimization of Jump; note that these analyzes gain more insight into the working principles of diversity mechanisms than into crossover itself. A breakthrough is Dang et al. [2018], showing how diversity emerges in the absence of diversity mechanisms.

A notable exception from this topical focus is the work by Sudholt [2017] on how crossover can speedup the optimization of the so-called OneMax function[1] by a factor of 2. This was a seminal result since it considered crossover in a more natural setting than Jump, which gives crossover an unfair advantage by placing the optimum where crossover frequently generates its output (see the criticism offered by Jansen [2015] on the (in-)significance of the Jump function). Similarly, Corus and Oliveto [2018] and Antipov et al. [2020] showed speedups for OneMax in a more intricate setting of crossover.

The work Doerr and Doerr [2015] considers a very different algorithm employing crossover, the $(1 + (\lambda, \lambda))$-GA. This algorithm, in each generation, produces a significant amount of diversity to be used by crossover. The interesting outcome is that, on OneMax, this algorithm is asymptotically faster than the $\Omega(n \log n)$ incurred by most other search heuristics on the OneMax test function.

In this work, we investigate a natural model for constrained optimization in evolutionary computation. Friedrich et al. [2020] suggested an extension of OneMax, which we call BoundMax$_B$, parametrized by the bound $B \in \mathbb{N}$ on the maximum allowed number of ones. This function is defined so that, for all $x \in \{0,1\}^n$ and with $|x|_1$ being the number of ones in x,

$$\text{BoundMax}_B(x) = \begin{cases} \sum_{i=1}^{B}(1+1/n)x_i + \sum_{i=B+1}^{n} x_i & \text{if } |x|_1 \leq B; \\ -|x|_1 & \text{otherwise.} \end{cases}$$

This linear function has a unique optimum with exactly B many ones in positions 1 through B. The first phase of the (1+1) EA[2] when maximizing BoundMax$_B$ typically collects 1-bits with only a slight preference for the first B positions (which we call *heavy*; in contrast to the other positions, which we call *light*). Once a solution at the cardinality bound B is found, it can only be improved by swapping out light bits for heavy ones. As shown by Friedrich et al. [2020], for some choice of

[1] OneMax maps a bit string to its number of 1-bits.

[2] The (1+1) *evolutionary algorithm* ((1+1) EA) maintains a best-so-far solution and, in each iteration, makes a copy of it, flips each bit independently with probability $1/n$ and keeps the new solution if it is at least as good as the previous one.

B, the (1+1) EA takes time $\Theta(n^2)$ to optimize this function. The given B was cn for some c with $1/2 < c < 1$, we extend this to values of $B = n - d$, where d is a constant.

Intuitively, the problem when optimizing at the constraint is that both a 0 and a 1 need to be flipped (in particular, for an almost optimal solution, a specific heavy 0 and a specific light 1). The first remedy that comes to mind is to use an operator that does exactly that. *Swap mutation* has been suggested in experimental works [Chen and Hou 2006; Meinl and Berthold 2009; Manzoni et al. 2020] alongside the *balanced crossover*, that is, crossover that keeps the same number of ones in the output. In a sense, this opens up the "black box" setting (which assumes no knowledge about the fitness function) into more of a "gray box", taking the specific problem knowledge into account that the number of ones in a solution carries importance. This is not only useful in our constrained setting, a special case of the knapsack problem, but also for other combinatorial problems that assign importance to the number of ones. For example, finding a minimum spanning tree requires finding exactly $n - 1$ edges, a minimal vertex cover tries to minimize the number of picked vertices, and so on. Understanding which general structural properties lead to better optimization performance on a wide range of different problems can thus be very beneficial [Whitley et al. 2016]. An operator swapping a 1 and a 0 would sacrifice the full unbiasedness[3] but maintains problem-specific unbiasedness [Rowe and Vose 2011]. Note that a balanced operator always needs to be paired with regular mutation, otherwise it is confined to only explore solutions with the same number of 1-bits.

We define swap mutation formally in Section 3.2 and give a rigorous analysis in Section 5. In particular, we show that it achieves an optimization time of $O(n \log(n))$ on $\textsc{BoundMax}_B$ for $B = n - d$ with d constant, intuitively being able to find a heavy 0 among constantly many 0s to swap easily (Theorem 5.5). However, for $B = cn$ with $1/2 < c < 1$, we get a lower bound of $\Omega(n^2)$ also for this operator, intuitively because there are a linear number of both 1- and 0-bits at the constraint.

We also use the regime of $B = cn$ with $1/2 < c < 1$ to investigate the effectiveness of crossover. (The bound $B = n - d$ for constant d gives a quadratic optimization time for all considered cases.) The use of crossover typically requires some kind of diversity, and while some diversity might emerge on its own [Dang et al. 2018], we employ specific diversity mechanisms. We start by considering *island models* [Neumann et al. 2011; Doerr et al. 2019], in particular the *single-receiver model* [Watson and Jansen 2007]. Mutation-based algorithms are run in parallel on *islands,* and their individuals are then considered for crossover at the single receiver. This maintains perfect independence of the individuals for maximum diversity.

We show that *uniform crossover*[4] with the single-receiver island model can now optimize $\textsc{BoundMax}_B$ in time $O(n^2/\log n)$, an asymptotic speedup over mutation-based algorithms. Roughly speaking, when only logarithmically many incorrect bits are left on each island, they are likely to be in different positions, and crossover finds the correct offspring in time exponential in the number of incorrect bits. Note that, from the birthday paradox, we know that if at each island the resident individual has only $O(\sqrt{n})$ many incorrect bits, then likely the islands have their incorrect bits at different positions. This motivates us to use a ternary operator, the deterministic *majority vote* [Friedrich et al. 2016], taking three parents and using the majority bit at each position for the offspring. Here, we can show a runtime bound of $O(n\sqrt{n})$ on $\textsc{BoundMax}_B$. Mutation reduces the number of incorrect bits at each island within $O(n\sqrt{n})$ iterations to be $O(\sqrt{n})$, then majority vote is successful. Note that we only provide upper bounds, so a final decision on how

[3] An operator is *unbiased* if, when given two problem instances where one is a perturbation—permutation of bit order and bit flips—of the other, it performs analogously [Lehre and Witt 2012].

[4] The output of the uniform crossover takes each bit independently from a parent chosen uniformly at random.

Table 1. Overview of Expected Runtimes on $\textsc{BoundMax}_B$

Algorithm Variant	$n - B = O(1)$	$n - B = \Theta(n)$	Reference
(1+1) EA	$\Theta(n^2)$	$\Theta(n^2)$	[Friedrich et al. 2020] Proposition 5.2
(1+1) Swap-EA	$O(n \log(n))$	$\Theta(n^2)$	Theorem 5.5 Theorem 5.6
(2+1) islands, (balanced) uniform	$\Theta(n^2)$	$\Theta\left(\frac{n^2}{\log(n)}\right)$	Theorem 6.12 Theorems 6.4 and 6.7
(3+1) islands, majority vote	$\Theta(n^2)$	$O(n\sqrt{n})$	Theorem 6.12 Theorem 6.8
(2+1) GA, balanced uniform, with diversity	$\Theta(n^2)$	$O(n \log(n))$	Theorem 6.12 Theorem 6.11
(2+1) Swap-GA, balanced uniform, with diversity	$O(n \log(n))$	$O(n \log(n))$	Theorem 6.14

The Swap-EA uses swap and standard mutation. Islands run the (1+1) EA and a single receiver with the stated crossover. The GA uses standard mutation and balanced uniform crossover, the Swap-GA additionally has swap mutation.

the two algorithms compare cannot be made. However, we believe that these bounds are tight, as mutation alone would be slower, and crossover can be applied only once to speedup optimization (since any offspring of crossover can never be the parent for another mutation nor crossover) in the single-receiver island model.

In order to mitigate the disadvantage of the single-receiver setting, we consider a **genetic algorithm (GA)** with the diversity mechanism of *Hamming-distance maximization* (but many others would serve equally well, see Dang et al. [2016]). We use an idea of Sudholt [2017] and show that a population of equally good individuals first spreads in Hamming distance and then gains in quality with crossover, making the process so fast that we obtain the desirable bound of $O(n \log n)$ for the optimization of $\textsc{BoundMax}_B$. Note that this improvement arises naturally from the interplay of crossover and mutation, and our analysis gives insight into this interplay.

Note that, for the last result, we employ a *balanced* uniform crossover: given two parents, the matching bits are inherited and exactly half of the remaining bits are set to 1. This operator produces the same number of 1s in the offspring as are in the parents. Using standard uniform crossover instead can delay optimization by producing offspring with the wrong number of ones.

The literature already discusses a wide range of balanced crossover operators [Chen and Hou 2006; Meinl and Berthold 2009; Manzoni et al. 2020], but we find that all of them lack other useful properties. They either do not bequeath matching bits or they are not order-unbiased (as defined in Lehre and Witt [2012]). To close this gap, we introduce the *balanced uniform crossover*, which is fully unbiased and has all the desired properties.

Table 1 gives a summary of our results. The remainder of this article is structured as follows. After fixing the notation in Section 2, we review crossover and mutation operators from the literature in Section 3. We introduce the balanced uniform crossover operator in Section 4 and then examine the optimization time for $\textsc{BoundMax}_B$ under balanced mutation (Section 5) and crossover (Section 6). We conclude in Section 7.

2 PRELIMINARIES

For some positive integer $n \in \mathbb{N}^+$, symbol $[n]$ stands for the set $\{1, \dots, n\}$. An array $A = [a_1, a_2, \dots, a_n]$ is a sequence of natural numbers (incl. 0). For $i, \ell, r \in [n]$, we use $A[i] = a_i$ to refer to its ith entry, and $A[\ell, r] = [a_\ell, a_{\ell+1}, \dots, a_r]$ for the subarray from indices ℓ to r (if $\ell > r$, then

$A[\ell, r]$ is empty). A bit string of length n is any element of $\{0,1\}^n$. We refer to its ith bit by x_i, so $x = x_1 x_2 \dots x_n$; similarly, for $M \subseteq [n]$, x_M denotes the bit string obtained by removing all bits with indices not in M, and $x[\ell, r]$ is the bit string $x_\ell x_{\ell+1} \dots x_r$. We use $|x|_1 = \sum_{i=1}^{n} x_i$ and $|x|_0 = n - |x|_1$ for the number of 1- and 0-bits of x, respectively. For two bit strings x, y, we denote their concatenation as xy. The strings with n 0- or 1-bits are 0^n and 1^n, respectively. A permutation is a bijective function $\sigma : [n] \to [n]$. We write $\sigma = [a_1, a_2, \dots, a_n]$ to refer to the permutation where, for every $i \in [n]$, $\sigma(i) = a_i$. For every permutation σ of $[n]$ and every bit string x, we define the permutation of x respective to σ as $\sigma_b(x) = x_{\sigma(1)} x_{\sigma(2)} \dots x_{\sigma(n)}$. For example, for $x = 10011$ and $\sigma = [3, 1, 5, 2, 4]$, we get $\sigma_b(x) = 01101$. An event occurs ***with high probability*** **(w.h.p.)** with respect to n, if it has probability at least $1 - 1/n^c$ for some constant $c > 0$.

3 BALANCED CROSSOVER AND MUTATION

A *crossover operator* is a potentially randomized algorithm that maps two parent bit strings $x, y \in \{0,1\}^n$ to an output bit string $z \in \{0,1\}^n$. We identify operators with their output distribution, where $p_A(z \mid x, y)$ denotes the probability that operator A samples z given inputs x, y. In this article, we investigate *balanced* crossover operators that retain the same number of ones as the input strings. We would also like to have certain additional regularity properties such as invariance under permutations (*order unbiasedness*[5]) and that, whenever the parents agree on a bit position, this bit is sampled in the output (*inheritance respect*[6]).

Definition 3.1 (Regularity Properties). A crossover operator A is

(1) *balanced* if $|x|_1 = |y|_1 \neq |z|_1$ implies $p_A(z \mid x, y) = 0$;
(2) *order-unbiased* if $p_A(z \mid x, y) = p_A(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y))$ holds for all permutations σ of $[n]$;
(3) *inheritance-respectful* if $x_i = y_i \neq z_i$ for some $i \in [n]$ implies $p_A(z \mid x, y) = 0$.

The idea of order unbiasedness is that the order of encoding the bits in the string should not matter for the algorithm. A crossover should be inheritance-respectful if it wants to combine strengths of different solutions (and leave creating novelty to mutation). This was studied in the literature as *geometric crossovers* [Moraglio and Poli 2004]: Crossovers are inheritance-respectful if and only if they are geometric crossovers under the Hamming distance metric.

3.1 Existing Crossover Operators

We review several crossover operators from the literature and classify them according to the regularity properties. We find that none of the existing crossovers are balanced, order-unbiased, and inheritance-respectful at the same time. We propose the *balanced uniform crossover* as an alternative in Section 4. (The *boring crossover* that returns some parent also satisfies all three properties.)

We compare nine crossover operators: the uniform, single- and two-point crossover from the survey by Katoch et al. [2021], the counter-based, zero lengths and map-of-ones crossover by Manzoni et al. [2020], the shrinking crossover of Chen and Hou [2006], and finally the balanced two-point crossover and alternating crossover[7] by Meinl and Berthold [2009]. Our results are summarized in Table 2. In Section 3.2, we briefly discuss balanced mutation.

Uniform, single-point, and two-point crossovers. The very common *uniform crossover operator* iterates over all bit positions and, for each of them, selects a random parent and takes its bit at this

[5] See Lehre and Witt [2012] for a discussion; they call this property "σ-invariance".
[6] See Radcliffe [1994]. He distinguishes respect (shared properties of the parents are inherited) and gene transmission (every property of an offspring is inherited from at least one parent). These concepts are equivalent when limited to bit strings. We use the more telling term *inheritance respect*.
[7] The alternating crossover is called "balanced uniform crossover" in the original work [Meinl and Berthold 2009]. We reserve this name for Section 4.

Table 2. Overview of Different Crossover Operators, Where a ✓ Denotes that the given Operator does have the Stated Property, and × that it does not

Crossover Operator	Balanced	Order-Unbiased	Inheritance-Respectful
Uniform	×	✓	✓
Single-Point	×	×	✓
Two-Point	×	×	✓
Counter-Based	✓	×	×
Zero Lengths	✓	×	×
Map-of-Ones	✓	✓	×
Shrinking	✓	×	✓
Balanced Two-Point	✓	×	×
Alternating	✓	×	✓
Boring	✓	✓	✓
Balanced Uniform	✓	✓	✓

position. The *single-point crossover* chooses a random position in $[n]$ and swaps all bits after that position between the parents. For swap position s, this results in the two strings $x[1,s]\,y[s{+}1,n]$ and $y[1,s]\,x[s{+}1,n]$. Without losing generality, we define as output the string that starts with the bits of x. The *two-point crossover* samples two indices and swaps the bit-range between them among the parents. Intuitively, none of those operators can be balanced as they take 1-bits from the parents regardless of their total number.

Throughout this section, we let s be an index drawn uniformly at random from $[n]$. For two indices that are independently and uniformly distributed in $[n]$, we use symbol ℓ to denote the smaller one and r for the larger.

Definition 3.2 (Uniform, Single-point, and Two-point Crossover). On input $x, y \in \{0,1\}^n$, the output string $z \in \{0,1\}^n$ is as follows.

(1) *Uniform crossover*: For every $i \in [n]$, bit z_i is uniformly distributed in $\{x_i, y_i\}$, independently of all other choices.
(2) *Single-point crossover*: $z = x[1, s{-}1]\,y[s,n]$.
(3) *Two-point crossover*: $z = x[1,\ell{-}1]\,y[\ell,r]\,x[r{+}1,n]$.

Lemma 3.3. *The uniform crossover is inheritance-respectful and order-unbiased, but not balanced.*

Proof. The operator is unbalanced since we have $p(00 \mid 01,10) = \frac{1}{4} > 0$. It is inheritance-respectful by definition. Furthermore, it is order-unbiased. For any permutation σ and input strings x, y, the bit at position $\sigma(i)$ is sampled from $x_{\sigma(i)}$ and $y_{\sigma(i)}$, yielding the same probability distribution over the permuted outcomes. □

We only show the proof for the single-point operator, the one for two points is essentially the same.

Lemma 3.4. *The single-point crossover is inheritance-respectful, but neither order-unbiased nor balanced.*

Proof. Inheritance respect is clear. The operator is unbalanced as $p(00 \mid 01,10) = \frac{1}{2} > 0$. It is order-biased: Let $n = 4, x = 0011, y = 1100, z = 1111$, and $\sigma = [1,3,2,4]$. We have $p(z \mid x,y) = \frac{1}{4}$, but $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = p(1111 \mid 0101, 1010) = 0$. □

Counter-based, zero lengths, and map-of-ones crossovers. To heal the unbalancedness of the uniform crossover, the *counter-based crossover* stops once the number of ones/zeros in the output are equal to that of the parent x.

Definition 3.5 (Counter-based Crossover). Inductively for all indices $i \in [n]$, let $z_{\leq i}$ be the string of bits already set after the ith iteration. The bit z_i is a uniform random value in $\{x_i, y_i\}$, independently of the previous choices. If $|z_{\leq i}|_1 = |x|_1$, the final output is $z = z_{\leq i}\, 0^{n-i}$; if $|z_{\leq i}|_0 = |x|_0$, the output is $z_{\leq i}\, 1^{n-i}$. Otherwise, the operator continues to the next iteration.

Lemma 3.6. *The counter-based crossover is balanced, but neither inheritance-respectful nor order-unbiased.*

Proof. The operator is balanced because it explicitly keeps track of the number of ones and zeros and, if this forces the remaining bits to be set in a certain way, does so. To show lack of inheritance respect, let $n = 3, x = 101, y = 011$, and $z = 110$. We thus have $p(110 \mid 101, 011) = 1/2 \cdot 1/2 = 1/4 > 0$ and $x_3 = y_3 \neq z_3$. Regarding the order bias, let additionally $\sigma = [3, 1, 2]$. We have that $p(z \mid x, y) = p(110 \mid 101, 011) = 1/4$ but $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = p(011 \mid 110, 101) = 0$, so p is not order-unbiased. □

For the other two operators, we need alternative ways to represent bit strings. The zero lengths representation is effectively a run-length encoding of consecutive 0-bits. Let $x \in \{0,1\}^n$ be such that $|x|_1 = k$ and $x = 0^{a_1}10^{a_2}1 \ldots 0^{a_k}10^{a_{k+1}}$ with $a_i \in \mathbb{N}_0$. Then, x's *zero lengths array* is $Z_x = [a_1, a_2, \ldots, a_k, a_{k+1}]$. The $k+1$ elements[8] of Z_x necessarily sum to $n-k$. Conversely, the *map-of-ones array* M_x contains the indices of all 1-bits in x. Any permutation of M_x also represents x, we tacitly use set notation if no ambiguity arises.

The *zero lengths crossover* constructs the zero lengths array of the output from that of the parents by picking each entry uniformly at random from the corresponding parent entries. It additionally ensures that the sum of elements does not exceed the number of zeros. The *map-of-ones crossover*, in each iteration, chooses a random parent string and samples one of its stored 1-indices that has not already been selected.

Definition 3.7 (Zero Lengths and Map-of-ones Crossover). Let $k = \min(|x|_1, |y|_1)$. The representation of output z is defined as follows.

(1) *Zero lengths crossover*: For all $i \in [k]$, let $s_{i-1} = \sum_{j=1}^{i-1} Z_z[j]$ be the sum of set entries. $Z_z[i] = \min(a_i, n-k-s_{i-1})$, with a_i uniformly distributed in $\{Z_x[i], Z_y[i]\}$; $Z_z[k+1] = n-k-s_k$.
(2) *Map-of-ones crossover*: For all $i \in [k]$, let u_i be uniformly distributed vector in $\{x, y\}$, entry $M_z[i]$ is set to a uniform random index in $M_{u_i}[1,k] \backslash M_z[1, i-1]$.

Lemma 3.8. *The zero lengths crossover is balanced, but neither inheritance-respectful nor order-unbiased.*

Proof. We note that this operator is balanced, as it creates an offspring whose zero lengths array is of the same size as the representation of the inputs, hence its number of ones matches the number of ones in the input. Also, it explicitly keeps track of the number of zeros and sets the last entry accordingly.

To show lack of inheritancy-respect, let $n = 3, x = 101, y = 011$, and $z = 110$, so $Z_x = [0,1,0], Z_y = [1,0,0], Z_z = [0,0,1]$.

[8] In Manzoni et al. [2020], the authors claim that the zero lengths array has $n-k+1$ entries. This is probably a typo, their number really is $k+1$.

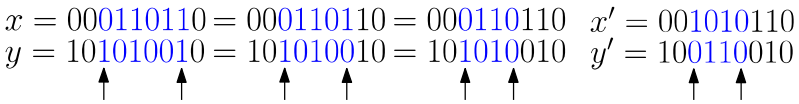

Fig. 1. Example run of the shrinking crossover operator. The initial range is shrunk twice. Then, the bits inside the range contain the same number of ones and are swapped.

We have that $p(z \mid x, y) = p([0,0,1] \mid [0,1,0],[1,0,0]) = 1/2 \cdot 1/2 \cdot 1 = 1/4 > 0$ and as additionally $x_3 = y_3 \neq z_3$, the operator is not inheritance-respectful.

Regarding order bias, let additionally $\sigma = [3,1,2]$. We have that $p(z \mid x, y) = p([0,0,1] \mid [0,1,0],[1,0,0]) = 1/2 \cdot 1/2 \cdot 1 = 1/4$ but $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = p(011 \mid 110, 101) = p([1,0,0] \mid [0,0,1],[0,1,0]) = 0$, so the operator is not order-unbiased. □

Lemma 3.9. *The map-of-ones crossover operator is balanced and order-unbiased, but not inheritance-respectful.*

Proof. This operator is balanced because it adds exactly k distinct elements to M_z.

Regarding inheritance respect, let $n = 3, x = 110, y = 011$, so we have $M_x = [1,2], M_y = [2,3]$. Consider the output $M_z = [1,3]$ representing the string $z = 101$. We have that $p(101 \mid 110, 011) = 2 \cdot 1/4 \cdot 1/4 > 0$, as the probability to first draw a 1 and then a 3, or to first draw 3 and then a 1, is $1/4 \cdot 1/4$ each. As additionally $x_2 = y_2 \neq z_2$, the operator is not inheritance-respectful.

Next, we prove the order-unbiasedness. For any permutation σ of $[n]$, we denote its inverse by σ^{-1}, that is, the unique permutation such that $\sigma(\sigma^{-1}(i)) = i$ for all $i \in [n]$. Recall that, after permuting x according to σ, the jth bit of the outcome is $(\sigma_b(x))_j = x_{\sigma(j)}$. A 1-bit x_i thus moves to position $j = \sigma^{-1}(i)$ in $\sigma_b(x)$. For the map-of-ones, that means $M_{\sigma_b(x)} = [\sigma^{-1}(M_x[1]), \dots, \sigma^{-1}(M_x[|x|_1])]$. The map-of-ones crossover operator neither considers the order in the map nor the numerical value of its elements. It only checks whether two elements are equal or not. For any string $z \in \{0,1\}^n$ and index $i \in [n]$, the operator samples $M_z[i]$ given x and y as inputs with the same (uniform) probability as $M_{\sigma_b(z)}[i] = \sigma^{-1}(M_z[i])$ is sampled given $\sigma_b(x)$ and $\sigma_b(y)$. Hence, $p(z \mid x, y) = p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y))$ and the operator is order-unbiased. □

Shrinking, balanced two-point, and alternating crossover. Both the shrinking and balanced two-point crossover operator are based on the regular two-point version. The *shrinking crossover* works on the classical bit string representation. Before swapping the sampled bit-ranges, the contained ones are counted. If they are not equal between the parents, the index r is reduced until they are (Figure 1). The reduction may result in $r = \ell - 1$, that is, an empty range, the output then equals x. The *balanced two-point crossover* applies the two-point operator to the map-of-ones representations of the parent strings. This may sample duplicate 1-indices. To recover balancedness, those are replaced by elements from the unchosen range of M_x. Finally, the *alternating crossover* also works on map-of-ones but is deterministic. It combines M_x, M_y in a sorted manner, keeping duplicates, and takes every other 1-index.

Below, let $A \diamond B$ denote the concatenation of arrays A and B.

Definition 3.10 (Shrinking, Balanced Two-point, and Alternating Crossover). Let $k = \min(|x|_1, |y|_1)$.

(1) *Shrinking crossover*: Let r' be the largest index $\ell \leq r' \leq r$ such that $|x[\ell, r']|_1 = |y[\ell, r']|_1$, or $r' = \ell - 1$ if there is none. The output string is $z = x[1, \ell{-}1]\, y[\ell, r']\, x[r'{+}1, n]$.
(2) *Balanced two-point*: Let u, v, with $u \leq v$, be independently and uniformly distributed in $[k]$ and $M = M_x[1, u{-}1] \diamond M_y[u, v] \diamond M_x[v{+}1, |x|_1]$. The output representation M_z is

generated from M by removing duplicates, sampling uniform random replacements from $M_x[u, v] \setminus M_z$.

(3) *Alternating crossover*: Let M be the result of sorting $M_x \diamond M_y$. The output is $M_z = M[1] \diamond M[3] \diamond \cdots \diamond M[2k-1]$.

Lemma 3.11. *The shrinking crossover operator is balanced and inheritance-respectful, but not order-unbiased.*

Proof. The operator is designed to be balanced, the inheritance respect follows as for the two-point crossover.

For the order bias, let $n = 4, x = 0011, y = 1100, z = 1001$, and $\sigma = [1, 3, 2, 4]$. We have that $p(z \mid x, y) = p(1001 \mid 0011, 1100) = 0$, as it would require to swap exactly the first and third bit.

However, $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = p(1001 \mid 0101, 1010) = 1/4$ by having the left bound at the first bit and the right bound at the second or third bit (and shrinking in the latter case). Thus, out of the 16 possible values for the two indices, (1,2), (2,1), (1,3), and (3,1) lead to 1001, giving the probability of 1/4. Hence, the operator is order-biased. □

Lemma 3.12. *The balanced two-point crossover operator is balanced, but neither inheritance-respectful nor order-unbiased.*

Proof. Regarding the operator's balance, provided that both inputs x, y have $|x|_1 = |y|_1 = k$, we know that $|z|_1 = k$ because M has exactly length k after initializing M, and every element that is deleted is replaced afterwards. We further note that the elements in M_x within the range suffice to replace the duplicates without adding new duplicates: We know that M_x contains k distinct values. All these elements outside of the range are already in M, and all elements within the range can be added if needed. This way we are guaranteed to find enough replacements to reach k many ones.

To show the lack of inheritance respect, let $n = 5, x = 11010, y = 01101$, and $z = 10101$. The respective map-of-ones are $M_x = [1, 2, 4]$, $M_y = [2, 3, 5]$, $M_z = [1, 3, 5]$. We have $p(10101 \mid 11010, 01101) = p([1, 3, 5] \mid [1, 2, 4], [2, 3, 5]) = 2/9 > 0$ (the probability to select a range from the second to the third map entry, so out of the nine possible values for the two sampled values, (2,3) and (3,2) succeed), but $x_2 = y_2 \neq z_2$.

Regarding order bias, we use the same n, x, y, and z as above, so once more $p(10101 \mid 11010, 01101) = 2/9$. However, given $\sigma = [1, 3, 4, 5, 2]$, we have $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = p(11010 \mid 10101, 01011) = 0$ as there is no way for the last map entry to be anything other than 5. So the operator is not order-unbiased. □

Lemma 3.13. *The alternating crossover operator is balanced and inheritance-respectful, but not order-unbiased.*

Proof. Regarding balancedness, given that both inputs x, y have $|x|_1 = |y|_1 = k$ their map-of-ones representations will each have length k. The combined array will therefore have length $2k$. Because that length is even, taking every odd index from the array will yield $2k/2 = k$ indices. The map-of-ones for the offspring consequently has length k. It also cannot contain duplicates: If it did, identical values would have been at non-adjacent positions in the combined array, and since each elements can appear twice at most, the array would be unsorted. The map-of-ones with length k and no duplicates guarantees that the derived bit string has exactly k many ones.

Regarding inheritance respect, let z be the result of an alternating crossover of two bit strings x and y of length n. We prove for each bit $1 \leq i \leq n$ that if $x_i = y_i = 1$ then $z_i = 1$. In this case i will appear twice in the combined array, and because the array is sorted, both occurrences of i are adjacent. One of them will be at an odd index in the array, and it will therefore be added to the map-of-ones of the offspring. Thus $z_i = 1$.

We further show for each bit that if $x_i = y_i = 0$ then $z_i = 0$. In this case i will not appear in the combined array at all. Since elements for the map-of-ones of the offspring are taken only from that array, i cannot appear in it. Thus $z_i = 0$.

We have therefore shown that $x_i = y_i$ implies $z_i = x_i$, proving the alternating crossover operator to be inheritance-respectful.

Regarding order bias, let $n = 5, x = 11010, y = 01101, z = 11010$. Per definition, $M_x = [1, 2, 4], M_y = [2, 3, 5], M_z = [1, 2, 4]$. As the combined and sorted array, we get $M_x \diamond M_y = [1, 2, 2, 3, 4, 5]$, of which the elements at every odd index are exactly $[1, 2, 4]$. Therefore $p(z \mid x, y) = 1$.

However, for $\sigma = [1, 2, 3, 5, 4]$, we have $\sigma_b(x) = 11001, \sigma_b(y) = 01110, \sigma_b(z) = 11001$. Accordingly, $M_{\sigma_b(x)} = [1, 2, 5], M_{\sigma_b(y)} = [2, 3, 4], M_{\sigma_b(z)} = [1, 2, 5]$. The combined array $M_{\sigma_b(x)} \diamond M_{\sigma_b(y)} = [1, 2, 2, 3, 4, 5]$ happens to be the same as before. Thus, $p(\sigma_b(z) \mid \sigma_b(x), \sigma_b(y)) = 0$, the operator is order-biased. □

3.2 Balanced Mutation

A *mutation operator* is an algorithm mapping a single parent bit string $x \in \{0, 1\}^n$ to an offspring $z \in \{0, 1\}^n$. The most common mutation operator is *standard bit mutation* (also called *uniform mutation*), flipping each bit of x independently with probability $\frac{1}{n}$. This does not preserve the number of ones. The asymmetric mutation operator from Neumann and Wegener [2007] and Jansen and Sudholt [2010], which independently flips each 0-bit of x with probability $\frac{1}{2|x|_0}$ and each 1-bit with probability $\frac{1}{2|x|_1}$, does so in expectation, but does not guarantee it. Adapting the notation of Definition 3.1, a mutation operator is *balanced* if $|z|_1 \neq |x|_1$ implies $p_A(z \mid x) = 0$. The literature discusses balanced mutation just very briefly [Chen and Hou 2006; Meinl and Berthold 2009; Manzoni et al. 2020]. The only notable operator is *swap mutation*. It chooses a 1- and a 0-bit of x uniformly at random and swaps them. (The strings 1^n and 0^n remain unchanged.) We use the name (1+1) *swap-evolutionary algorithm* ((1+1) Swap-EA) for the variant of the (1+1) EA that, in the mutation step, uses the swap operator with probability p_b and standard bit mutation otherwise.

4 NEW BALANCED CROSSOVER OPERATOR

None of the crossovers discussed are balanced, order-unbiased, and inheritance-respectful. Of course, one could achieve all this by using what we call the *boring crossover* that merely returns one of the parents (say, at random), but it defeats the purpose of recombining the strengths of multiple individuals. We propose the *balanced uniform crossover*[9] operator instead. To ensure inheritance respect, it first fixes all positions in which the input strings are equal. For the others, it randomly samples as many positions as required to match the number of ones in the input strings.

Definition 4.1 (Balanced Uniform Crossover). Define the sets $F = \{i \in [n] \mid x_i = y_i\}$ and $\overline{F} = [n] \backslash F$. Let the set $I \subseteq \overline{F}$ with $|I| = \lfloor |\overline{F}|/2 \rfloor$ be drawn uniformly at random. The output bit string z has $z_i = x_i$ for all $i \in F$, $z_i = 1$ for $i \in I$, and $z_i = 0$ for $i \in \overline{F} \backslash I$.

Theorem 4.2. *The balanced uniform crossover operator is balanced, inheritance-respectful, and order-unbiased.*

Proof. Regarding balancedness, consider the substrings $x_{\overline{F}}$ and $y_{\overline{F}}$ in which the parents disagree. The 1-bits in $x_{\overline{F}}$ are in one-to-one correspondence to the 0-bits in $y_{\overline{F}}$, whence $|x_{\overline{F}}|_1 = |\overline{F}| - |y_{\overline{F}}|_1$. If additionally $|x|_1 = |x_F|_1 + |x_{\overline{F}}|_1$ and $|y|_1 = |y_F|_1 + |y_{\overline{F}}|_1$ are equal, we have

[9] As mentioned, we discuss what is called "balanced uniform crossover" in Meinl and Berthold [2009] as the alternating crossover.

$|x_{\overline{F}}|_1 = |y_{\overline{F}}|_1$. Then, $|\overline{F}|$ must be even and $|x_{\overline{F}}|_1 = |\overline{F}|/2$. Both things together imply $|z|_1 = |z_F|_1 + |I| = |x_F|_1 + |\overline{F}|/2 = |x|_1$.

The only other property that is possibly in doubt is the order bias. Let σ be a permutation of $[n]$ and F_σ the set of indices on which the perturbed strings $\sigma_b(x), \sigma_b(y)$ agree, $\overline{F_\sigma}$ its complement. Note that $i \in F$ holds if and only if $\sigma(i) \in F_\sigma$, in particular, $|F| = |F_\sigma|$ and $|\overline{F}| = |\overline{F_\sigma}|$. This means that for every $i \in [n]$, the probability of setting $z_i = 1$ given x, y is equal to the probability of setting $z_{\sigma(i)} = 1$ given $\sigma_b(x), \sigma_b(y)$ and order unbiasedness follows. □

The next theorem gives an alternative distributional characterization of the balanced uniform crossover.

Theorem 4.3. *The output distribution of the balanced uniform crossover is the same as that of the uniform crossover conditioned on having $|z|_1 = \lfloor (|x|_1 + |y|_1)/2 \rfloor$ 1-bits in the output.*

Proof. For all bit positions in F, the output bits of the balanced uniform crossover are the same as that of its regular counterpart, this holds even without the condition. For indices $i \in \overline{F}$, the operators differ in general. For the uniform crossover, $z_i = 1$ and $z_i = 0$ have the same probability. In other words, the set of 1-bits in that part of the output is a uniform random subset of $\overline{F}$. When conditioning on $|z|_1 = \lfloor (|x|_1 + |y|_1)/2 \rfloor$, this subset must have size

$$\left\lfloor \frac{|x|_1 + |y|_1}{2} \right\rfloor - |z_F|_1 = \left\lfloor \frac{|x_{\overline{F}}|_1 + |y_{\overline{F}}|_1}{2} \right\rfloor = \left\lfloor \frac{|\overline{F}|}{2} \right\rfloor$$ □

5 OPTIMIZING WITHOUT CROSSOVER

Here, we first show that the (1+1) EA has a quadratic lower bound on $\textsc{BoundMax}_B$, even if $n - B = O(1)$. For this case, we then break this bound by introducing balanced mutation: The (1+1) Swap-EA has an expected runtime in $O(n \log(n))$. For the case $n - B = \Theta(n)$, however, the runtime remains in $\Omega(n^2)$. We formalize both algorithms with Algorithm 1, where we obtain the standard (1+1) EA for $p_b = 0$ and the (1+1) Swap-EA for $0 < p_b < 1$. In Section 6, we show how crossover can improve optimization time in this case.

ALGORITHM 1: (1+1) (Swap-)EA

```
x ← uniform choice from {0,1}^n
while x is not optimum for f do
    if p_b > uniform choice from [0,1] then
        x' ← swap mutation(x)
    else x' ← standard uniform mutation(x)
    if f(x') ≥ f(x) then x ← x'
```

Theorem 5.1 ([Friedrich et al. 2020], Theorem 10). *There exists a constant $0 < c < 1$ such that the* (1+1) EA *using only standard bit mutations takes expected time $\Omega(n^2)$ to optimize* $\textsc{BoundMax}_B$ *under uniform constraint $B = cn$.*

It is easy to verify that their proof remains valid for $B = cn$ for *all* constants $0 < c < 1$ and, in fact, even for $B = n - n^c$. Below, we extend this to the case that the bound differs from the maximum n only by an *additive* constant, which is not covered by Friedrich et al. [2020].

Proposition 5.2. *For every positive integer constant d, the* (1+1) EA *using only standard bit mutation takes expected time $\Omega(n^2)$ to optimize* $\textsc{BoundMax}_B$ *under uniform constraint $B = n - d$.*

Proof. We first consider the case of $d = 1$ and accordingly consider $B = n - 1$. Recall that the bit positions with weight $1 + 1/n$ are "heavy" and others "light", so, in this case, we have exactly one light bit. Our first goal is to show that, when the algorithm reaches the bound of B for the first time, the light bit is set to 1 with probability at least $1/2$.

To this end, let X_t be the value of the light bit after t iterations of the algorithm and let T be the first time that the algorithm samples a search point with at least B many 1s. We show, by induction on t,

$$\forall t \leq T : \Pr[X_t = 1] \geq 1/2.$$

This holds trivially for $t = 0$. Let now $t < T$ be given with $\Pr[X_t = 1] \geq 1/2$. We abbreviate $p = \Pr[X_t = 1]$. Let C_u be the event that the algorithm flips strictly more *heavy* bits from 0s to 1s than vice versa (going *up*), C_e the event that the flip of the algorithm flips the same number of heavy bits from 0 to 1 as vice versa (*equal*), and C_d that more bits are flipped from 0 to 1 than vice versa (*down*).

We have

$$\Pr[X_{t+1} = 1] = p(1 - \Pr[C_u]/n) + (1 - p)\Pr[C_u \cup C_e]/n$$

by considering the two possible cases for X_t in the law of total probability and seeing that flipping a light 0 bit up to 1 is rejected exactly in case of C_d, and a flip of a light bit from 1 to 0 is accepted exactly in case of C_u.

We use $q_u = \Pr[C_u]$, $q_e = \Pr[C_e]$ and $q_d = \Pr[C_d]$ as abbreviations. Thus, we now have

$$\begin{aligned}\Pr[X_{t+1} = 1] &= p(1 - q_u/n) + (1 - p)(q_u + q_e)/n = p - pq_u/n + q_u/n + q_e/n - pq_u/n - pq_e/n \\ &= p + (1 - 2p)q_u/n + (1 - p)q_e/n.\end{aligned}$$

First we consider the case $p \geq 1/2 + 1/n$. From $(1 - 2p) \geq -1$, we get $(1 - 2p)q_u/n \geq -1/n$, giving

$$p + (1 - 2p)q_u/n + (1 - p)q_e/n \geq p - 1/n \geq 1/2.$$

We now consider the case $p < 1/2 + 1/n$.

Before we can proceed, we consider the probability that a specific *heavy* bit is 1; by symmetry, in any given iteration, this probability is the same for all bits (and by separability of the problem, the value of the different bits are stochastically independent). This probability starts with $1/2$ at initialization. Focusing on the optimization of the heavy bits only, we see that the algorithm proceeds as if the (1+1) EA was optimizing OneMax on $n - 1$ bits with mutation probability of $1/n$: Flipping the light bit can only change acceptance of the offspring if there is no impact on the heavy bits (only neutral steps might be discarded), which is not relevant for the optimization of OneMax. Thus, the probability that a specific heavy bit is 1 only increases over time, and is thus always at least $1/2$.

Consider now an iteration of the algorithm in which it flips a set S of heavy bits. We match any bit pattern that the current bit string might have on the bits S with its complement and see that the one of the two where at least as many 1s as 0s are flipped is at least as likely as the other. This gives $q_u \leq q_d$, and, in particular, $q_u \leq (1 - q_e)/2$.

We continue the above equations using $q_u \leq (1 - q_e)/2$.

$$\begin{aligned}&= p + (1 - 2p)q_u/n + (1 - p)q_e/n \geq p + (1 - 2p)(1 - q_e)/(2n) + (1 - p)q_e/n \\ &\geq p - \frac{2}{n}(1 - q_e)/(2n) + (1/2 - 1/n)q_e/n = p - \frac{1}{n^2} + \frac{1}{2n}q_e.\end{aligned}$$

We have that $q_e \geq 1/e$, since the event that no heavy bit flips contributes to C_e. This shows that the last expression in the derivation above is at least p for $n \geq 2e$, as desired.

Now we know that, when the (1+1) EA for the first time matches (or exceeds) the bound of B, the probability that the light bit is 1 is at least 1/2. Since, for any bit flip pattern that changes the current bit string into the all-1 bit string (exceeding B) there are many more likely bit flip patterns that do not exceed B, we get that, with constant probability, the first bit string with at least B bits has exactly B bits. Thus, in this case, the solution is suboptimal and only one particular 2-bit-flip can improve the fitness; the expected time for this 2-bit-flip to happen is $\Omega(n^2)$ as desired. The case of larger (but still constant) d follows from the observation that for more light bits it is overall more likely to have at least one light bit set to 1, which means that still with constant probability the first solution at the boundary is not optimal, and still there are only constantly many 2-bit-flips which can improve. Also, it is still much more likely to reach the bound B exactly than to overshoot it. □

Recall that the (1+1) Swap-EA applies swap mutation in each round with probability p_b. Next, we prove that it outperforms the (1+1) EA if $n - B = O(1)$. To this end, we start with showing that the expected optimization time from the point on where the first individual with B 1-bits is sampled is $O(B(n-B))$.

Lemma 5.3. *For any $0 < p_b < 1$, consider the* (1+1) Swap-*EA on* BoundMax$_B$ *with uniform constraint B. Assume that the initial individual has exactly B 1-bits and let T be the random variable describing the number of steps until the algorithm finds the optimal solution. Then $E[T] \leq \frac{B(n-B)\pi^2}{6p_b}$.*

Proof. We use the fitness level method to derive the upper bound. For this, consider the fitness-based partition $A_0, \ldots, A_B$ of $\{0,1\}^n$ where A_i is the set of bit strings that have exactly i heavy 1-bits. Note that this partition is valid since the number of 1-bits in an offspring can neither increase nor decrease compared to its parent as we assume the process to start with an individual with exactly B ones and thus a mutation that changes the number of 1-bits would decrease fitness and thus be rejected.

Let p_i be a lower bound on the probability of jumping to a fitness level above A_i assuming that the current individual is in A_i. Then, the fitness level method allows us to estimate an upper bound for $E[T]$ as $E[T] \leq \sum_{i=0}^{B-1} \frac{1}{p_i}$ (cf. [Wegener 2002, Section 8 Lemma 1]).

We show that the probability p_i is lower-bounded by $p_i \geq p_b \frac{(B-i)^2}{B(n-B)}$. For this, consider a bit string $x \in A_i$ recall that it has i heavy 1-bits. The swap mutation operator converts x to a higher fitness level if it swaps one of the $B-i$ heavy 0-bits and one of the $B-i$ light 1-bits. There are B 1-bits and $n-B$ 0-bits, since we start with an individual with B 1-bits and cannot lose 1-bits as this would decrease fitness. Hence, an improving swap happens with probability $\frac{B-i}{B} \cdot \frac{B-i}{n-B} = \frac{(B-i)^2}{B(n-B)}$. Since the swap mutation operator is invoked with probability p_b, we obtain $p_i \geq p_b \frac{(B-i)^2}{B(n-B)}$.

Using the fitness level method, this means $E[T]$ is at most

$$\sum_{i=0}^{B-1} \frac{1}{p_i} \leq \frac{B(n-B)}{p_b} \sum_{i=0}^{B-1} \frac{1}{(B-i)^2} = \frac{B(n-B)}{p_b} \sum_{i=1}^{B} \frac{1}{i^2} \leq \frac{\pi^2 B(n-B)}{6p_b}$$

since the series $\sum_{i=1}^{\infty} \frac{1}{i^2}$ converges to $\pi^2/6$ (the Basel problem). □

For the time until the first individual with B 1-bits is sampled, we get the following upper bound. The proof extends the argument of Friedrich et al. [2020, Theorem 9] to the notion that using balanced mutation with constant probability only adds a constant factor.

Lemma 5.4. *For any $0 \leq p_b < 1$, consider the* (1+1) Swap-*EA on* BoundMax$_B$ *with uniform constraint B. Let T be the random variable describing the number of steps until the algorithm finds a solution with exactly B 1-bits. Then, $E[T]$ is in $O(n \log(n))$.*

Proof. This is shown for the standard (1+1) EA on OneMax under uniform constraint B in Friedrich et al. [2020, Theorem 9]. This proof is analogous for the optimization of the function $\textsc{BoundMax}_B$ since any mutation flipping only a single 0-bit is accepted as long as the number of ones is strictly smaller than B. If the number of ones is larger than B, any mutation flipping only a single 1-bit is accepted. The runtime of our algorithm that additionally uses balanced mutation is the same asymptotically since the balanced mutation does not change the number of ones and hence only contributes a constant factor to the runtime. □

Combining Lemmas 5.3 and 5.4 now yields the following bounds.

Theorem 5.5. *For any $0 < p_b < 1$, consider the* (1+1) Swap-*EA on* $\textsc{BoundMax}_B$. *Let T be the random variable describing the number of steps until the algorithm finds the optimum. Assume that p_b is constant and that $B = n - d$. If $d = O(\log(n))$, then $E[T]$ is in $O(n \log(n))$. If $d = \Omega(\log(n))$, $E[T]$ is in $O(nd)$.*

Proof. The expected time until the first solution with exactly B 1-bits is obtained is in $O(n \log(n))$ by Lemma 5.4.

From Lemma 5.3, it follows that the expected time until the optimum is reached from that point is at most $\frac{B(n-B)\pi^2}{6p_b} = \frac{d(n-d)\pi^2}{6p_b}$. For $d = O(\log(n))$, this term is in $O(n \log(n))$, yielding a total expected optimization time of $O(n \log(n))$. For $d = \Omega(\log(n))$, the term is in $O(nd)$, which gives a total expected optimization time of $O(n \log(n) + nd) = O(nd)$. □

Next, we show that the runtime bound of $O(n^2)$ for the case of $B = cn$ is tight by giving a matching lower bound. The proof is similar to that of Friedrich et al. [2020, Theorem 10], which treated the case $p_b = 0$. We show that for the focal points of the analysis swap mutation does not help. In more detail, we prove that w.h.p. either $\Omega(n^2)$ iterations pass without finding the optimum or the process samples an individual with a constant, positive Hamming distance to the optimum. Fixing the last missing positions then takes quadratic time.

Theorem 5.6. *For any constant $0 \leq p_b \leq 1$, consider the* (1+1) Swap-*EA on* $\textsc{BoundMax}_B$. *Let T be the random variable describing the number of steps until the algorithm finds the optimum. There is a constant $0 < c < 1$ such that, for $B = cn$, it holds that $E[T] = \Omega(n^2)$.*

Proof. We lean on the proof of Theorem 10 in Friedrich et al. [2020], e.g., we consider the same bound $c = \frac{3}{4}$, whence the constraint is $B = \frac{3}{4}n$. The (1+1) EA samples an initial solution with at most $\frac{2}{3}n$ 1-bits w.h.p. due to a Chernoff bound. If $p_b = 1$, then the (1+1) Swap-EA *never* reaches the optimum x^* because a balanced operator cannot increase the number of 1-bits. We can thus assume $p_b < 1$.

Note that the Hamming distance between the initial solution and the x^* is linear in n. We claim that w.h.p. either $\Omega(n^2)$ iterations pass or a solution with Hamming distance between 4 and 8 to the optimal solution is sampled. Let d denote that Hamming distance. In their proof, Friedrich et al. [2020] have shown that the probability of directly jumping from an individual with $d > 8$ to a solution with $d < 4$ is in $O(\frac{1}{n^6})$ when employing only the standard mutation operator ($p_b = 0$). We note that any application of the swap mutation is also unable to skip the range, since it can reduce d by 2 at most. The probability of skipping the range thus remains in $O(1/n^6)$ in our case. The probability that no such jump happens in the first $\Omega(n^2)$ iterations is therefore at least $1 - O(1/n^4)$.

We continue our analysis at the first point in time at which a solution with Hamming distance between 4 and 8 is sampled. Either the individual already has exactly B 1-bits or we can apply the following argument to show that it reaches exactly B 1-bits before reaching the optimum w.h.p.

To this end, we prove that after $r = \frac{2\ln(n)}{1-p_b}$ iterations the sampled solution will have B many 1-bits w.h.p. but has still not reached the optimum. Again, only standard mutation can increase the number of 1s. By once more employing a Chernoff Bound argument, we get that among r iterations w.h.p. at least $\ln(n)$ standard mutation steps will happen. From here, we again refer to Friedrich et al. [2020], where the probability for reaching exactly B many 1-bits after $\ln(n)$ standard mutations is proven to be $1 - \frac{1}{n^{\Omega(1)}}$.

Conversely, we prove that w.h.p. the optimal solution x^* is not sampled within these $r = \frac{2\ln(n)}{1-p_b}$ iterations for any constant p_b. Let $4 \leq d \leq 8$ be the number of wrong bits in the current solution x, i.e., the Hamming distance between x and x^*. To sample x^*, at least these d bits have to change their value at least once. Let p_{flip} be an upper bound on the probability of flipping[10] any specified bit in an iteration. Then the probability to never flip a certain bit in r iterations is at least $(1 - p_{\text{flip}})^r$. Now the probability to not flip all of the d incorrect bits in r iterations is at least $1 - (1 - (1 - p_{\text{flip}})^r)^d$. When standard bit mutation is applied, we get $p_{\text{flip}} \geq \frac{1}{n}$. For swap mutation, the probability of flipping a 1-bit is at most $\frac{1}{B}$, and the probability of flipping any specific 0-bit is at most $\frac{1}{n-B} = \frac{1}{n-cn} = \frac{4}{n}$, which is greater than $\frac{1}{n}$ and $\frac{1}{B} = \frac{4}{3n}$. Hence, we safely assume $p_{\text{flip}} = \frac{4}{n}$. By Bernoulli's inequality, we get that the probability of not flipping all d wrongly set bits is

$$1 - (1 - (1 - p_{\text{flip}})^r)^d \geq 1 - \left(1 - \left(1 - \frac{4}{n}\right)^r\right)^d \geq 1 - \left(\frac{4r}{n}\right)^d$$
$$\geq 1 - \left(\frac{8\ln(n)}{(1-p_b)\cdot n}\right)^4 \geq 1 - \frac{8^4\ln(n)^4}{(1-p_b)^4\cdot n^4} = 1 - O\left(\frac{1}{n^3}\right).$$

We have established that the algorithms either runs for $\Omega(n^2)$ iterations or reaches a solution that has exactly B 1-bits but is not optimal. Namely, it has a Hamming distance to x^* between 4 and 8. This part even holds w.h.p. To show $E[T] = \Omega(n^2)$ over all, it suffices to prove that from this point to an optimal solution requires expected quadratic time. Note that in the current situation we have two to four incorrect heavy 0-bits and the same number of incorrect light 1-bits. We assume that every mutation of the bit string that reduces the Hamming distance is accepted. For standard bit mutation to achieve this, it must flip at least one of the heavy 0-bit and one light 1-bit at the same time. The chances for this event are maximum if there are four incorrect bits to choose in each block, but even then the probability is at most $\frac{16}{n^2}$. Similarly, regarding swap mutation the chance of an improvement is at most $\frac{4}{B} \cdot \frac{4}{n-B} = \frac{16}{cn(n-cn)} = \frac{256}{3n^2}$. In both cases, even a single improvement already requires an optimization time in $\Omega(n^2)$. □

We have hence shown that swap mutation is superior to balanced mutation for $\textsc{BoundMax}_B$ at least if $n - B = O(1)$ since then the number of possible outcomes of swap mutation is small. However, this advantage vanishes if $n - B = \Theta(n)$ (Theorem 5.6) as this results in a significantly higher number of possible mutation outcomes.

6 OPTIMIZATION WITH CROSSOVER

We are interested whether crossover can break the quadratic barrier for $\textsc{BoundMax}_B$. For this, we analyze different scenarios. We start with a (2 + 1) single-receiver island model as described in Watson and Jansen [2007] with unbalanced as well as balanced uniform crossover. We then extend our analysis to a (3 + 1) single-receiver island model with majority vote crossover [Friedrich et al. 2016].

[10]Throughout this proof we use the more common term "flipping" to indicate the change of a bit value, regardless of whether it is by standard bit mutation or swapping.

ALGORITHM 2: $(\mu + 1)$ Single-Island Receiver Model

$x_1, \ldots, x_\mu, y \leftarrow$ independent, uniform choices from $\{0,1\}^n$
while y *is not optimum for* f **do**
  foreach x_i **do**
    $x_i' \leftarrow \text{mutate}(x_i)$
    if $f(x_i') \geq f(x_i)$ **then** $x_i \leftarrow x_i'$
  $x_{i_1}, x_{i_2} \leftarrow$ independent, uniform choices from $\{x_1, \ldots, x_\mu\}$
  $y' \leftarrow \text{crossover}(x_{i_1}, x_{i_2})$
  if $f(y') \geq f(y)$ **then** $y \leftarrow y'$

In the $(\mu + 1)$ single-receiver island model, μ instances of the (1+1) EA are running independently on their own island using standard mutation. After each iteration, crossover is applied to two randomly selected residents of these μ islands. The offspring replaces the resident on the receiver island if it has at least the same fitness. See Algorithm 2 for a formalization. In the case of majority vote crossover, three instead of two out of μ individuals are chosen. Our analysis focuses on the cases $\mu = 2$ and $\mu = 3$, respectively, as it would not benefit from a larger population.

After analyzing the island models, we consider the $(\mu + 1)$ GA with balanced uniform crossover and a diversity mechanism (Algorithm 3). In this setting, a population of μ individuals is maintained whereby in each iteration, one offspring is created. This happens with probability p_c by means of crossover on two randomly chosen individuals, and otherwise by means of standard mutation on a randomly selected individual. Out of the $\mu + 1$ individuals at the end of each iteration, one individual with lowest fitness is removed. Ties are broken according to a rule aimed at increasing diversity. We use *Hamming-distance maximization*, where an individual is chosen such that the sum of the pairwise Hamming distances of the remaining population is maximized. We refer to Dang et al. [2016, Section 4.2] for details. Again, we focus on $\mu = 2$.

At the end of the section, we analyze a variant of the (2+1) GA that incorporates both balanced (uniform) crossover and balanced (swap) mutation. We call it the (2+1) Swap-GA. Here, in each iteration, crossover is applied with probability p_c and otherwise it performs balanced mutation with probability p_b and unbalanced mutation else.

A summary of our results can be found in Table 1. It turns out that for cardinality constraints that satisfy $n - B = \Theta(n)$, crossover provides polynomial gains over the performance of the (1+1) Swap-EA, all the way down to an expected running time of $O(n \log(n))$ for the (2+1) GA as well as the (2+1) Swap-GA. If, however, B is additively close to n (i.e., $n - B = O(1)$), then we show that not even the (2+1) GA can be better than quadratic. This is in stark contrast to the (1+1) Swap-EA optimizing those instances in time $O(n \log(n))$ without any crossover (Theorem 5.5). The (2+1) Swap-GA, which combines the advantages of both balanced crossover and mutation, indeed maintains its superior performance also for $n - B = O(1)$.

6.1 (2+1) Island Model with (Balanced) Uniform Crossover

We show that the runtime is reduced from $O(n^2)$ for the (1+1) Swap-EA to $O(n^2/\log(n))$ in this setting. For this, we divide the optimization process into two stages. First we wait until the individuals on all non-receiver-islands have B 1-bits and only a certain number of heavy 0-bits. This step is captured in Lemma 6.1. Then, we analyze the time it takes the crossover to sample the optimum given that we have passed the first stage. We note that our fitness function ensures that we never go back to the first stage.

ALGORITHM 3: (2+1) GA. Selection Ties are Broken by Maximizing the Hamming Distance.

```
x_1, x_2 ← independent, uniform choices from {0,1}^n
while neither x_1 nor x_2 is optimum for f do
    if p_c > uniform choice from [0,1] then
        y ← crossover(x_1, x_2)
    else
        x ← uniform choice from {x_1, x_2}
        y ← mutate(x)
    {x_1, x_2} ← maximum-fitness subset of {x_1, x_2, y}
```

Lemma 6.1. *Consider the* (1+1) EA *using only standard mutations on* $\textsc{BoundMax}_B$ *starting with an individual with exactly B 1-bits. Let T be the random variable describing the number of steps until the algorithm finds a solution with at most* $k \in \mathbb{N}^+$ *heavy 0-bits. Then,* $E[T]$ *is in* $O(\frac{n^2}{k})$.

Proof. We find an upper bound by employing the fitness level method with fitness levels $A_{\leq k}, A_{k+1}, \ldots, A_B$, where A_i is the set of bit strings with i heavy 0-bits and $A_{\leq k}$ contains all bit strings with at most k such bits. Note that the partition is not over all bit strings of length n but only over those with exactly B 1-bits as we assume that the optimization starts with one such individual. The initial solution is at worst in A_B, and we are interested in the time until a solution in $A_{\leq k}$ is first sampled.

Let p_i be the probability of leaving the fitness level A_i in an iteration. We show that $p_i \geq \frac{i^2}{en^2}$. To leave the level A_i, it suffices to flip one of the i heavy 0-bits as well as one of the i light 1-bits while leaving all other bits unchanged. The probability for this event is $\frac{i}{n} \cdot \frac{i}{n} \cdot (1-\frac{1}{n})^{n-2} \geq \frac{i^2}{en^2}$.

We now apply the fitness level method to find the desired upper bound. Let T' be the random variable denoting the number of iterations to reach level $A_{\leq k}$. We get

$$E[T'] \leq \sum_{i=k+1}^{B} \frac{1}{p_i} \leq \sum_{i=k+1}^{B} \frac{en^2}{i^2} = en^2 \sum_{i=k+1}^{B} \frac{1}{i^2}.$$

Then, we use the integral to bound the sum. Because $\frac{1}{i^2}$ is strictly decreasing and we decrease the lower end by one, we get a valid upper bound of

$$E[T'] \leq en^2 \int_{k+1-1}^{B} i^{-2}\, di = en^2[-1 \cdot i^{-1}]_k^B = en^2\left(-\frac{1}{B} + \frac{1}{k}\right).$$

This implies a total runtime in $O(\frac{n^2}{k})$. □

In order for crossover to be able to create the optimal solution, we need the individuals x and y on the first two islands to be free of so-called *blocking bits*. We define a blocking bit as a position i such that $x_i = y_i = 0$ if $i \leq B$ and $x_i = y_i = 1$ if $i > B$. Intuitively, a blocking bit is a position for which both individuals differ from the optimal solution. By counting the possibilities to remove a blocking bit by means of flipping two bits, one gets an estimate for the probability to remove blocking bits.

Lemma 6.2. *Let* $x, y \in \{0,1\}^n$ *be two individuals that both have exactly B 1-bits where* $B = cn$ *for some constant* $0 < c < 1$. *Assume further that both x and y have at most* $i = o(n)$ *heavy 0-bits and there is at least one blocking bit. The probability that a standard mutation on one of x and y removes a blocking bit without decreasing fitness is at least* $\frac{\min(c,1-c)}{2en}$ *for large enough n.*

Proof. We consider the event that a blocking bit is removed by only swapping the blocking bit with another bit in the same block of the chosen individual x. If the blocking bit is a heavy bit of x, the bit that it is swapped with must be at a position where both x and y have a 1-bit as otherwise the number of blocking bits would remain unchanged. Analogously, if the blocking bit is light, its swap partner must be at a position for which both individuals are 0. We call such bits *suitable* bits. Since x and y have at most i heavy 0-bits and hence at most i light 1-bits each, there are at least $B - 2i = cn - 2i$ suitable heavy bits and $n - B - 2i = (1-c)n - 2i$ suitable light bits. The probability p of removing a blocking bit hence becomes

$$p \geq \frac{\min(c, 1-c)n - 2i}{n^2}\left(1 - \frac{1}{n}\right)^{n-2} \geq \frac{\min(c, 1-c)}{en} - \frac{2i}{en^2}$$

since $i = o(n)$, we have $\frac{2i}{en^2} = o(\frac{1}{n})$ and hence $p \geq \frac{\min(c,1-c)}{2en}$ for large enough n. □

Next, we analyze the probability that, assuming the parent individuals are free of blocking bits, balanced uniform crossover samples the optimal solution.

Lemma 6.3. *Let $x, y \in \{0,1\}^n$ be two individuals without blocking bits that both have exactly B 1-bits and Hamming distance h. For any $z \in \{0,1\}^n$, the sampling probability $p(z \mid x, y)$ of the balanced uniform crossover is either 0 or $\Theta(\sqrt{h}/2^h)$.*

Proof. Note that for each heavy bit position i on which x and y agree, we have $x_i = y_i = 1$ as otherwise there would be blocking bits. Let k be the number of heavy bits in which the individuals differ. Since both have B 1-bits, the number of differing light bits is also k. Each position in the light block for which one of the individuals is 1 is a differing position as otherwise there would be blocking bits. We thus have $h = 2k$ for the Hamming distance. Among those h positions, both x and y have k 1-bits and k 0-bits.

The balanced uniform crossover chooses exactly k 1-bits at the differing positions with each of the $\binom{h}{k}$ possible outcomes being equally likely. We lower-bound the probability $1/\binom{2k}{k}$ via Stirling's approximation as $\sqrt{\pi k}/2^{2k}$ via the following computation.

To lower-bound the probability $1/\binom{2k}{k}$, we give an upper bound for $\binom{2k}{k} = (2k)!/(k!)^2$. Using Stirling's approximation, we get

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n+1}} \leq n! \leq \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}.$$

This yields

$$\binom{2k}{k} \leq \frac{\sqrt{4\pi k}\left(\frac{2k}{e}\right)^{2k} e^{\frac{1}{24k}}}{\left(\sqrt{2\pi k}\left(\frac{k}{e}\right)^k e^{\frac{1}{12k+1}}\right)^2} = \frac{2^{2k}}{\sqrt{\pi k}} \cdot e^{\frac{1}{24k} - \frac{2}{12k+1}} \leq \frac{2^{2k}}{\sqrt{\pi k}}.$$

The last step uses $\frac{1}{24k} - \frac{2}{12k+1} \leq 0$. As $k = h/2$, we get that the crossover finds any specific solution with probability at least $\sqrt{\frac{\pi}{2}}\frac{\sqrt{h}}{2^h}$. □

With this, we derive the following upper bound for the $(2+1)$ single-receiver island model. It is independent of whether the balanced uniform crossover or the regular variant is used. The idea is to first estimate the time until both individuals have B 1-bits and at most $\log_2(n/4)$ heavy 0-bits by using Lemmas 5.4 and 6.1. Then, drift analysis is employed on a potential function reflecting the number of blocking bits as well as the probability of the crossover sampling the optimum given that there are no more blocking bits, by employing Lemmas 6.2 and 6.3.

Theorem 6.4. *The expected optimization time of the* (2+1) *single-receiver island model using (balanced) uniform crossover for* $\textsc{BoundMax}_B$ *with* $B = cn$ *for any constant* $0 < c < 1$ *is in* $O(\frac{n^2}{\log(n)})$.

Proof. We analyze three stages separately. The first one is the runtime until both EA islands have an individual with B 1-bits and at most $\log_2(n/4)$ heavy 0-bits. Secondly, we look at the time until there are no blocking bits, and the third stage is the time until the balanced uniform crossover samples the optimal solution from individuals without blocking bits.

From Lemmas 5.4 and 6.1, we get that each EA island produces an individual with B 1-bits and at most $\log_2(n/4)$ heavy 0-bits in $O(n\log(n) + \frac{n^2}{\log(n)})$ iterations. The time until both EA islands have reached this point is at most twice as long, yielding the same asymptotic runtime.

Assuming two individuals with at most $\log_2(n)/4$ heavy 0-bits, we now employ drift analysis over the number of blocking bits to analyze how long it takes until there are no blocking bits left. Let therefore X_t be the potential in iteration t. We define the potential to be 0 if the optimal solution is on the receiver island. Otherwise, the potential is

$$X_t = i \cdot \frac{2en}{\min(c, 1-c)} + n,$$

where i is the number of blocking bits. We upper-bound the expected change in potential for each iteration, given i.

Negative drift. The potential increases only if a 1-bit and a 0-bit within a block of one individual are flipped where the other individual has the respective inverted bit as follows. If a bit is the same in both parents, swapping it with another bit would at best create a blocking bit while also removing a blocking bit. Swapping bits between blocks would be rejected if a blocking bit was created because the fitness decreases. Because both individuals have at most $\log_2(n)/4$ heavy 0-bits and $\log_2(n)/4$ light 1-bits, respectively, either block of one parent has at most $\log_2(n)/4$ bits that can be swapped. The probability of this event is at most $\frac{\log^2(n)}{16n^2}$ and it increases the potential by at most $\frac{2en}{\min(c,1-c)}$ since either the 1-bit or the 0-bit in a block can cause a blocking bit, not both. It is also possible that multiple bits are swapped at once, but this probability decreases rapidly and can be accounted for with a constant factor γ. Therefore, the resulting negative drift is $\frac{2en}{\min(c,1-c)} \cdot \gamma\frac{\log^2(n)}{16n^2} = o(1)$ and approaches 0 for large n.

Positive drift for $i > 0$. The potential decreases by $\frac{2en}{\min(c,1-c)}$ for each blocking bit that is removed. If there is any blocking bit, we get from Lemma 6.2 that a blocking bit is removed with probability at least $\frac{\min(c,1-c)}{2en}$. The overall positive drift for $i > 0$ is therefore at least 1.

Positive drift for $i = 0$. For the last stage, we now lower bound the probability of creating the optimal solution in the crossover given that there are no blocking bits. At this point, individuals with at most $\log_2(n)/4$ heavy 0-bits have been sampled on both islands and any individual with more 0-bits has been rejected. With at most $\log_2(n)/4$ heavy 0-bits, both current individuals have at most $\log_2(n)/2$ bits that differ from the optimal solution. They therefore differ from one another in at most $\log_2(n)$ bits. We get from Lemma 6.3 with $m = \log_2(n)$ that balanced uniform crossover samples the optimal solution at this point with a probability of at least $\sqrt{\frac{\pi}{2}}\frac{\sqrt{\log_2(n)}}{2^{\log_2(n)}} = \sqrt{\frac{\pi}{2}}\frac{\sqrt{\log_2(n)}}{n}$. For (unbalanced) uniform crossover, the optimal solution is sampled with probability $1/2^m$. Hence, we get for large enough n, that the probability of sampling the optimal solution for both operators is at least $1/2^m = 1/n$. Since the potential reduces by n when the optimal solution is sampled, this results in a positive drift of at least 1.

Hence, there is a constant $\delta > 0$ such that the drift $E[X_t - X_{t+1} \mid X_t = s] \leq \delta$. By applying the additive drift theorem, we therefore get an expected runtime of at most $\delta \cdot (\log_2(n)\frac{2en}{\min(c,1-c)} + n)$ when starting with solutions that have at most $\log_2(n)$ blocking bits. This is in $O(n\log(n))$. The overall runtime is then $O(\frac{n^2}{\log(n)} + n\log(n)) = O(\frac{n^2}{\log(n)})$. □

We further prove an identical lower bound. Lemma 6.6 gives a lower bound in cases where the first solution with B 1-bits has few 0-bits. Theorem 6.7 derives a general lower bound. The following lemma bounds the probability v_i required for the fitness level method.

Lemma 6.5. *Consider the* (1+1) EA *using only standard mutation on* $\textsc{BoundMax}_B$ *with cardinality constraint* $B = cn$ *for some constant* $c > 0$. *Then the following two statements hold:*

(i) If the optimization starts with an individual with exactly B 1-bits and b heavy 0-bits, then, for all $0 \leq i < b$, the probability v_i that there is an iteration such that the current individual has exactly i heavy 0-bits is at least $1 - \frac{ei^2}{2n^2}$.

(ii) If the optimization starts with an individual with $b > B$ 1-bits, then the probability v that there is an iteration such that the current individual has fewer than B 1-bits before the first individual with exactly B 1-bits is sampled is at most $\frac{eB}{n}$.

Proof. For the first statement, we define the fitness levels $A_0, A_1, \ldots, A_B$, where A_i is the set of bit strings of length n with i heavy 0-bits and B 1-bits. Note that these fitness levels partition the set of all individuals reachable in the process since no individual with less than B 1-bits is ever accepted. We further use the notation $A_{\leq i}$ to describe the set $A_0 \cup A_1 \cup \ldots \cup A_i$.

We denote by $A_j \to A_{\leq i}$ the event that the process jumps, in one iteration, from the current individual being in A_j to a state in which the individual is in $A_{\leq i}$. As shown by Doerr and Kötzing [2021, Lemma 3.10], if we can prove for a $q \in [0,1]$ that it satisfies $\Pr[A_j \to A_i \mid A_j \to A_{\leq i}] \geq q$ for all $j > i$, then q is a valid lower bound for the probability v_i. Equivalently, if we find a q' such that $\Pr[A_j \to A_{\leq i-1} \mid A_j \to A_{\leq i}] \leq q'$, then this implies $v_i \geq 1 - q'$.

We therefore estimate $\Pr[A_j \to A_{\leq i-1} \mid A_j \to A_{\leq i}]$. Note that

$$\Pr[A_j \to A_{\leq i-1} \mid A_j \to A_{\leq i}] = \frac{\Pr[(A_j \to A_{\leq i-1}) \cap (A_j \to A_{\leq i})]}{\Pr[A_j \to A_{\leq i}]} = \frac{\Pr[A_j \to A_{\leq i-1}]}{\Pr[A_j \to A_{\leq i}]}.$$

In order to jump from A_j with $j > i$ to $A_{\leq i}$, at least $j - i$ heavy 0-bits need to be "swapped", i.e., flipped at the same time, with $j - i$ light 1-bits in a single iteration. The probability that this occurs for a fixed set of $j - i$ heavy 0-bits and $j - i$ light 1-bits is $n^{-2(j-i)}$. A union bound over all choices of such pairs of sets gives $\Pr[A_j \to A_{\leq i}] \leq \binom{j}{j-i}^2 n^{-2(j-i)}$.

On the other hand, swapping any set of $j - i$ heavy 0-bits with any set of $j - i$ light 1-bits and not changing any other bit is sufficient to jump from A_j to $A_{\leq i}$, so

$$\Pr[A_j \to A_{\leq i}] \geq \binom{j}{j-i}^2 \frac{1}{n^{2(j-i)}} \left(1 - \frac{1}{n}\right)^{n-2(j-i)} \geq \binom{j}{j-i}^2 \frac{1}{2en^{2(j-i)}}.$$

In conclusion,

$$\Pr[A_j \to A_{\leq i-1} \mid A_j \to A_{\leq i}] \leq \frac{\binom{j}{j-i+1}^2 n^{-2(j-i+1)}}{\binom{j}{j-i}^2 (2en^{2(j-i)})^{-1}} = \frac{\binom{j}{j-i+1}^2 2en^{2(j-i)}}{\binom{j}{j-i}^2 n^{2(j-i+1)}} = \frac{2ei^2}{(j-i+1)^2 n^2} \leq \frac{ei^2}{2n^2},$$

where the last step holds for all $i < j$ as then $j - i + 1 \geq 2$.

For the second statement, we instead define the fitness level A_i as the set of bit strings with exactly i 1-bits for all $0 \leq i \leq n$. To bound the probability of ever jumping from a level A_j with $j > B$

to a level with $i < B$, we note that for any $j > B$, $\Pr[A_j \to A_{\leq i}] \leq \binom{j}{j-i} n^{-(j-i)}$ which follows by an analogous argument as above, now the $j - i$ bits only need to be flipped not swapped. Similarly, we obtain for any $j > B$, $\Pr[A_j \to A_{\leq i}] \geq \binom{j}{j-i} n^{-(j-i)}/(2e)$ and thus

$$\Pr[A_j \to A_{\leq i-1} \mid A_j \to A_{\leq i}] \leq \frac{\binom{j}{j-i+1} 2en^{j-i}}{\binom{j}{j-i} n^{j-i+1}} = \frac{2ei}{(j-i+1)n} \leq \frac{ei}{n}.$$

Inserting $i = B$ concludes the proof. □

In the next lemma, we use the above findings to derive a lower tail bound on the number of steps needed to reduce a solution with exactly B 1-bits and b heavy 1-bits to a solution with a heavy 1-bits. Our method is similar to that used by Witt [Witt 2014]. However, the bounds in Witt [2014] are not applicable directly to our case so we use a modified version.

Lemma 6.6. *Consider the* (1+1) EA *using only standard mutation on* $\textsc{BoundMax}_B$ *with cardinality constraint* $B = cn$*, for some constant* $c > 0$*, optimizing an individual with exactly* B *1-bits and* b *heavy 0-bits. Let* T *be the random variable describing the number of steps until the algorithm finds a solution with at most* a *heavy 0-bits. If* $b = o(n)$*,* $a \geq 1$*, and* $b = \omega(a)$*, then* $E[T] \geq (1 - o(1))\frac{n^2}{a+1}$ *and*

$$\Pr\left[T \leq \frac{n^2}{2e^2(a+1)}\right] \leq \exp\left(-\frac{a}{8e}\right) + \frac{eb^3}{2n^2}.$$

Proof. To bound the expectation of T, we employ the fitness level method for lower bounds from Doerr and Kötzing [2021]. We define the fitness levels $A_{\leq a}, A_{a+1}, \ldots, A_B$, where A_i is the set of n-bit strings with exactly B 1-bits and i heavy 0-bits. Set $A_{\leq a}$ contains all bit strings with exactly B 1-bits and at most a heavy 0-bits. Again, the levels partition the relevant search space.

We are interested in the time until a solution in $A_{\leq a}$ is sampled for the first time. Let p_i be the probability of leaving the fitness level A_i in one iteration. Because at least one heavy 0-bit and one light 1-bit need to be flipped simultaneously for that, we get $p_i \leq \frac{i^2}{n^2}$.

Furthermore, let v_i be the probability of ever visiting level A_i when starting in A_b. Lemma 6.5 (i) gives $v_i \geq 1 - \frac{ei^2}{n^2}$ which is lower bounded by $1 - o(1)$ for all $i \leq b = o(n)$. The fitness level method implies $E[T] \geq \sum_{i=a+1}^{b} \frac{v_i}{p_i} \geq (1 - o(1)) \sum_{i=a+1}^{b} \frac{n^2}{i^2}$. We use an integral to bound the sum.

$$\begin{aligned} E[T] &\geq (1 - o(1))n^2 \int_{a+1}^{b} x^{-2}\,\mathrm{d}x = (1 - o(1))n^2[-x^{-1}]_{a+1}^{b} \\ &= (1 - o(1))n^2\left(-\frac{1}{b} + \frac{1}{a+1}\right) = (1 - o(1))\left(1 - \frac{a+1}{b}\right)\frac{n^2}{a+1} = (1 - o(1))\frac{n^2}{a+1}, \end{aligned}$$

where we used $b = \omega(a)$.

We now turn to the probability bound in the second part of the lemma. Recall that we derived above that the probability of skipping level A_i is at most $\frac{ei^2}{2n^2}$. We show that likely none of the levels A_i for $a \leq i \leq b$ are skipped. We denote this event by $\mathcal{E}$. Conditioned on $\mathcal{E}$, the time spent in each of these levels is an independent geometric random variable. We subsequently apply a Chernoff bound on their sum. First note that, by a union bound, we get that the probability of skipping any of the levels A_i for $a \leq i \leq b$ is

$$\Pr[\overline{\mathcal{E}}] \leq \sum_{i=a}^{b} \frac{ei^2}{2n^2} \leq \frac{eb^3}{2n^2}.$$

Denote by T_i the (random) time spent in level A_i. Conditioned on $\mathcal{E}$, each T_i (for $a \leq i \leq b$) is an independent geometric random variable with success probability $p_i \leq \frac{i^2}{n^2}$. Indeed, if level A_i is

visited, the number of steps spent there and the index of the next level visited after A_i are in fact independent. Thus, conditioning on the event that the level visited after A_i is A_{i-1} for all i does not influence the number of time spent in each level.

Accordingly, conditioned on $\mathcal{E}$, we have that $T = \sum_{i=a}^{b} T_i$ where each T_i is an independent geometric random variable and $E[T \mid \mathcal{E}] = \sum_{i=a}^{b} \frac{1}{p_i} \geq \sum_{i=a}^{b} \frac{n^2}{i^2} = (1 - o(1))\frac{n^2}{a+1}$ as shown above. Let $p_{\min} = \min\{p_1, p_2, \ldots, p_n\}$. We refer to Janson [Janson 2018, Theorem 3.1], who showed that if X is a sum of independent geometric variables X_i with success probability p_i, then for any $\lambda < 1$ we have

$$\Pr[X \leq \lambda E[X]] \leq \exp\left(-p_{\min} E[X](\lambda - 1 - \ln(\lambda))\right).$$

In our case $p_{\min} \geq \frac{a^2}{2en^2}$ and $E[T \mid \mathcal{E}] \geq \frac{n^2}{2(a+1)}$, thus choosing $\lambda = e^{-2}$ gives

$$\Pr\left[T \leq \frac{n^2}{2e^2(a+1)} \mid \mathcal{E}\right] \leq \exp\left(-\frac{a^2}{4e(a+1)}(e^{-2} - 1 + 2)\right) \leq \exp\left(-\frac{a}{8e}\right),$$

where we used that $a \geq 1$, whence $a + 1 \leq 2a$. Let $\mathcal{T}$ abbreviate the event that $T \leq n^2/2e^2(a+1)$.

$$\begin{aligned}\Pr[\mathcal{T}] &= \Pr[\mathcal{T} \mid \mathcal{E}] \cdot \Pr[\mathcal{E}] + \Pr[\mathcal{T} \mid \overline{\mathcal{E}}] \cdot \Pr[\overline{\mathcal{E}}] \leq \Pr[\mathcal{T} \mid \mathcal{E}] + \Pr[\overline{\mathcal{E}}] \\ &\leq \exp\left(-\frac{a}{8e}\right) + \frac{eb^3}{2n^2}.\end{aligned}$$

□

Using the last two preparatory lemmas, we now present a lower bound on the expected running time of our island model that matches the upper bound in Theorem 6.4. Note that we prove Theorem 6.7 below only for balanced uniform crossover while Theorem 6.4 also holds for the unbalanced version. We believe that the lower bound can be generalized but this may involve a more tedious case distinction.

Theorem 6.7. *There exists a constant $0 < c < 1$ such that the expected optimization time of the (2+1) single-receiver island model using balanced uniform crossover for* $\textsc{BoundMax}_B$ *with cardinality constraint $B = cn$ is in $\Omega(\frac{n^2}{\log(n)})$.*

Proof. We choose $c = 1/8$ and show that there is at least a constant probability that the optimal solution is not sampled within the first $rn^2/\log(n)$ iterations, where r is a constant to be fixed later.

Note that balanced crossover can only sample the optimal solution if both parent individuals have exactly B 1-bits. We show that with constant probability on both non-receiver islands the first individual with exactly B 1-bits has a linear number of heavy 0-bits. We then use the tail bound from Lemma 6.6 to show that in the following $rn^2/\log(n)$ iterations (again with constant probability), no individual with at most $\log(n)$ heavy 0-bits is sampled. We finally show that, under this condition, it is sufficiently unlikely that crossover creates the optimum solution in this phase.

Fix some non-receiver island. We claim that with constant probability, the individual on the island does not achieve a state with exactly B 1-bits and at most $\log(n)$ heavy 0-bits within the first $rn^2/\log(n)$ iterations. By a Chernoff bound, we get that the first sampled individual has more than $B = n/8$ 1-bits w.h.p. Then, by Lemma 6.5 (ii), the probability of ever jumping from a state with $j > B$ 1-bits to a state with $i < B$ 1-bits before the first individual with exactly B 1-bits is sampled is at most $\frac{eB}{n} = \frac{e}{8}$. Hence, with probability at least $(1 - o(1))(1 - e/8)$, the first individual with exactly B 1-bits on our island is obtained before the first individual with strictly less than B 1-bits is sampled. In this case, no bias regarding the number of heavy 1-bits can occur and thus the first time our individual has exactly B 1-bits, these bits are distributed uniformly within the bit string. By another Chernoff bound, the number of heavy 0-bits is thus linear in n w.h.p.

Now we get from Lemma 6.5 (i) that the probability of never obtaining an individual with $i = \sqrt{n}$ heavy 0-bits when starting with linearly many heavy 0-bits is at most $\frac{e}{2n} = o(1)$. We thus get that

with probability at least $(1-o(1))(1-e/8)$, there is an iteration in which our individual has $\sqrt{n}$ heavy 0-bits. We show that from this point forward, with probability $1-o(1)$, the number of heavy 0-bits after $rn^2/\log(n)$ iterations is still greater than $\log(n)$. Denoting by T the (random) number of steps until there are at most $a = \log(n)$ heavy 0-bits when starting at $b = \sqrt{n}$ heavy 0-bits, we get from the tail bound in Lemma 6.6 that

$$\Pr\left[T > \frac{n^2}{2e^2(\log(n)+1)}\right] \geq 1 - \exp\left(\frac{-\log(n)}{8e}\right) - \frac{e}{2\sqrt{n}} = 1 - o(1).$$

In total, with probability at least $(1-o(1))(1-e/8)$, our individual still has more than $\log(n)$ heavy 0-bits after $rn^2/\log(n)$ iterations, where $r = \frac{1}{3e^2}$. Accordingly, the probability that this does not occur on one island is in $\Omega(1)$ and, as both islands are independent, with probability in $\Omega(1)$ it occurs on both islands.

Conditioned on this event, we show that the probability that balanced crossover samples the optimal solution within these first $rn^2/\log(n)$ iterations is $o(1)$. Again, the probability that crossover samples the optimum is 0 until both individuals on the non-receiver islands have exactly B 1-bits. The first time both these individuals have exactly B 1-bits, we may now assume that for the following $rn^2/\log(n)$ iterations, both individuals have at least $\log(n)$ heavy 0-bits and thus their Hamming distance is at least $4\log(n)$ assuming that they are free of blocking bit positions (positions at which both individuals have a heavy 0-bit or a light 1-bit), which we may freely assume as otherwise the probability of sampling the optimum by means of crossover is 0 due to the inheritance respectfulness of our operator. Thus, by Lemma 6.3, the probability that the optimum is not sampled within the first $rn^2/\log(n)$ iterations is at least

$$\left(1 - \frac{\sqrt{4\log(n)}}{n^4}\right)^{rn^2/\log(n)} \geq \left(1 - \frac{1}{n^3}\right)^{n^2} = 1 - o(1). \qquad \square$$

6.2 (3+1) Island Model with Majority Vote Crossover

We further reduce the optimization time to $O(n\sqrt{n})$ by instead using the deterministic majority vote crossover as introduced in Friedrich et al. [2016]. This operator requires three parent individuals and sets each bit in the offspring to the value that the majority of the parents exhibits at the respective position.

Theorem 6.8. *The expected optimization time for* $\textsc{BoundMax}_B$ *with* $B = cn$ *for constant* $0 < c < 1$ *in the* (3+1) *single-receiver island model with majority vote crossover operator is in* $O(n\sqrt{n})$.

Proof. We first wait until all EA islands have sampled individuals with B 1-bits. By Lemma 5.4, this takes parallel time $O(n\log(n))$.

Let T be the random variable describing the number of iterations until the optimum is sampled starting with three individuals with B 1-bits. By rewriting the definition of the expected value, we have

$$E[T] = \sum_{t=1}^{\infty} \Pr[T \geq t] \leq n\sqrt{n} \cdot \sum_{r=0}^{\infty} \Pr[T \geq rn\sqrt{n}]. \tag{1}$$

Note that we switched from the number of iterations t to the number r of cycles of $n\sqrt{n}$ iterations. Furthermore, $\Pr[T \geq 0] = 1$ and $\Pr[T \geq t+1] = \Pr[T \geq t] \cdot (1 - \Pr[T = t \mid T \geq t])$. Hence,

$$\Pr[T \geq t+1] = \prod_{i=1}^{t}(1 - \Pr[T = i \mid T \geq i]) \leq 1 - \Pr[T = t \mid T \geq t].$$

In order to estimate $\Pr[T = t \mid T \geq t]$, we define $\mathcal{S}$ as the event that majority vote crossover succeeds in creating the optimum in a single step, and $\mathcal{E}_k$ that there are at most k wrong bits in each of the two blocks of each individual. The occurrence of $\mathcal{S}$ given $\mathcal{E}_k$ holds means that necessarily no wrongly set bit is shared by two or more individuals. Suppose these errors inside each block are uniformly distributed and independent from the previous cycle, then

$$\begin{aligned}\Pr[\mathcal{S} \mid \mathcal{E}_k] &\geq \frac{\binom{B-k}{k}}{\binom{B}{k}}\frac{\binom{B-2k}{k}}{\binom{B}{k}} \cdot \frac{\binom{n-B-k}{k}}{\binom{n-B}{k}}\frac{\binom{n-B-2k}{k}}{\binom{n-B}{k}} \\ &\geq \left(1 - \frac{6k^2}{B}\right) \cdot \left(1 - \frac{6k^2}{n-B}\right) = 1 - \frac{6k^2 n + 36k^4}{(c - c^2)n^2},\end{aligned}$$

because of the following reasons. The majority vote crossover succeeds if there is no bit position where at least two individuals have an incorrect value. For the first block of B bits, the second individual does not share an incorrect bit with the first individual with probability at least $\binom{B-k}{k}/\binom{B}{k}$. Similarly, third individual does not share an incorrect bit with neither the first nor second individual with probability at least $\binom{B-2k}{k}/\binom{B}{k}$. Using an analogous estimation for the second block of $n - B$ bits yields the first inequality. The second inequality employs the same estimate as used in Friedrich et al. [2016, Theorem 3.3].

By Lemma 6.1, we get that the expected number of heavy 0-bits after $r \cdot n\sqrt{n} - 1$ iterations is at most $c'\sqrt{n}/r$ for a constant c' and large enough n. As each individual has B 1-bits, the number of light 1-bits is the same. Furthermore, we have $\Pr[T = t \mid T \geq t] \geq \Pr[\mathcal{S} \mid \mathcal{E}_k]$ with k being the maximum number of wrongly set bits in each block after t iterations. Hence, there is a constant c'' such that $\Pr[T = rn\sqrt{n} - 1 \mid T \geq rn\sqrt{n} - 1]$ is at least

$$1 - \frac{6(c'/r)^2 + 36(c'/r)^4}{(c - c^2)} \geq 1 - \frac{c''}{r^2}.$$

We get $\Pr[T \geq rn\sqrt{n}] \leq c''/r^2$ and by inserting this back in Equation (1) and once more applying the Basel problem, we get

$$E[T] \leq n\sqrt{n} \sum_{r=0}^{\infty} \frac{c''}{r^2} \leq n\sqrt{n} \cdot \frac{c''\pi^2}{6}.$$

It remains to argue that w.h.p., at the end of each cycle, the wrongly set bits are uniformly distributed and independent from the previous cycle. This holds for the first cycle, as all weights inside both blocks are equal, and standard mutation is order-unbiased. For all other cycles it holds w.h.p., as there are sufficiently many correctly set bits such that the probability of a wrongly set bit being swapped to another position by flipping a correctly and and incorrectly set bit is in $O(\frac{1}{n})$, yielding w.h.p. each wrongly set bit position is swapped inside its block at least twice each cycle. The desired uniformity and independence follows by the following argument. From the second cycle on, we know that per block there are $O(\sqrt{n})$ wrongly set bits. In particular, there is a constant α such that there are at least αn correctly set bits in each block. Hence, the probability of swapping any specific wrongly set bit to another position is at least

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

The probability of swapping each wrongly set bit at least once during half a cycle is hence at least

$$\left(1-\left(1-\frac{\alpha}{2en}\right)^{n\sqrt{n}/2}\right)^{c'\sqrt{n}} \geq \left(1-\left(\left(1-\frac{\alpha}{2en}\right)^{n}\right)^{\sqrt{n}/2}\right)^{c'\sqrt{n}} \geq \left(1-\left(e^{-\alpha/(2e)}\right)^{\sqrt{n}/2}\right)^{c'\sqrt{n}}$$
$$= \left(1-e^{-\alpha\sqrt{n}/(2e)}\right)^{c'\sqrt{n}} \geq 1-e^{-\alpha\sqrt{n}/(2e)}c'\sqrt{n} \geq 1-\frac{1}{n^{1/2}}$$

where we employed Bernoulli's Inequality for the third step, and where the last line holds for large enough n. Hence, w.h.p. each wrongly set bit is swapped once during half a cycle, so w.h.p. every wrongly set bit is swapped two times each cycle, yielding a uniform distribution and independence from the previous cycle. □

6.3 (2+1) GA with Hamming-Distance Maximization

We show that we can reduce the runtime to $O(n\log(n))$ by employing the (2+1) GA and balanced uniform crossover. The crucial step in our analysis is to consider the event that the algorithm can reduce the number of wrongly set bits not only by means of mutation, but also by means of crossover, since the offspring produced by crossover is not put on an individual island, but instead competes with the other individuals directly. The probability of making progress by such an event is in fact constant if there is at least one non-blocking bit in each block.

Lemma 6.9. *Let $x, y \in \{0,1\}^n$ be two individuals with exactly B 1-bits and exactly i heavy 1-bits. Assume that there are $2a$ heavy bit positions and $2b$ light bit positions ($a, b \geq 1$) at which x and y differ. The probability that balanced uniform crossover samples a solution with more than i heavy 1-bits is at least $\frac{1}{6}$ and thus in $\Omega(1)$.*

Proof. Let X be the random variable denoting the number of heavy 1-bits in the offspring z. We want to find a lower bound on the probability $\Pr[X > i]$. For this, we note that $\Pr[X > i] = \Pr[X < i]$ since the number of possible outcomes of the crossover having more than a of the $2a$ heavy bit positions set to 1 is exactly equal to the number of possible outcomes with less than a of the $2a$ heavy bit positions set to 1. This is true since each bit string with less than a 1-bits can be obtained by inverting a bit string with more than a 1-bits.

Hence, we note that $2\Pr[X > i] + \Pr[X = i] = 1$. We show that the probability $\Pr[X = i]$ is at most $\frac{2}{3}$, which implies that the probability $\Pr[X > i]$ is at least $1/6$. For this, we note that $X = i$ if exactly half of the $2a$ differing positions in the first block are set to 1. This even has probability

$$\Pr[X = i] = \frac{\binom{2a}{a}\binom{2b}{b}}{\binom{2a+2b}{a+b}}$$

since there are $\binom{2a}{a}\binom{2b}{b}$ possible outcomes with exactly a of $2a$ differing heavy bit positions set to 1 and $\binom{2a+2b}{a+b}$ possible outcomes of the crossover in total. We show that the above expression is maximal for $a = b = 1$ where it reaches a value of $\frac{\binom{2}{1}\binom{2}{1}}{\binom{4}{2}} = \frac{2}{3}$. For this, we simplify as follows:

$$\Pr[X = i] = \frac{\binom{2a}{a}\binom{2b}{b}}{\binom{2a+2b}{a+b}} = \frac{(2a)!(2b)!}{(a!)^2(b!)^2}\frac{((a+b)!)^2}{(2a+2b)!} = \frac{(2a)!(2b)!}{(2a+2b)!}\left(\frac{(a+b)!}{a!b!}\right)^2.$$

We denote the last term with $g(a,b)$ and show that it is strictly monotonically decreasing in both a and b. For this, we show that $g(a+1,b) < g(a,b)$.

$$\begin{aligned} g(a+1,b) &= \frac{(2a+2)!(2b)!}{(2a+2b+2)!}\left(\frac{(a+b+1)!}{(a+1)!b!}\right)^2 = g(a,b)\frac{(2a+2)(2a+1)}{(2a+2b+2)(2a+2b+1)}\left(\frac{a+b+1}{a+1}\right)^2 \\ &= g(a,b)\frac{2(a+1)(2a+1)(a+b+1)^2}{2(a+b+1)(2a+2b+1)(a+1)^2} = g(a,b)\frac{(2a+1)(a+b+1)}{(2a+2b+1)(a+1)} \\ &= g(a,b)\frac{2a^2+2ab+3a+b+1}{2a^2+2ab+3a+2b+1} < g(a,b), \end{aligned}$$

since the denominator of the last fraction is strictly greater than the numerator. In the same way, we can show that $g(a,b+1) < g(a,b)$. Therefore, we get that $\Pr[X=i]$ is at most $2/3$ for all $a,b \geq 1$. Hence $\Pr[X>i]$ is at least $\frac{1}{6}$. □

In contrast to that, we note that (unbalanced) uniform crossover has a strictly worse probability of achieving improvement that is sub-constant and decreases with the number of positions at which two individuals differ. We use Stirling's approximation and exploit that even the probability of maintaining the number of 1-bits is less than constant.

Proposition 6.10. *Let $x,y \in \{0,1\}^n$ be two individuals with exactly B 1-bits and exactly i heavy 1-bits. Assume that there are $2a$ heavy bit positions and $2b$ light bit positions ($a,b \geq 1$) at which x and y differ. The probability that uniform crossover samples a solution with B many 1-bits and more than i heavy 1-bits is in $O(1/\sqrt{a+b})$.*

Proof. Let again X be the random variable denoting the number of heavy 1-bits in the offspring z. Let further $\mathcal{E}$ be the event that z has exactly B 1-bits. The probability p that uniform crossover samples a solution with more than i heavy 1-bits but exactly B 1-bits is thus

$$p = \Pr[X > i \mid \mathcal{E}]\Pr[\mathcal{E}] \leq \Pr[\mathcal{E}].$$

The probability of $\mathcal{E}$ is equal to the number of bit strings with exactly B 1-bits divided by the total number of possible outcomes of the crossover operator, i.e., $\Pr[\mathcal{E}] = \binom{2a+2b}{a+b}/2^{2a+2b}$. Using Stirling's approximation like in Lemma 6.3, we get that

$$\Pr[\mathcal{E}] = \frac{\binom{2a+2b}{a+b}}{2^{2a+2b}} \leq \frac{2^{2a+2b}}{\sqrt{\pi(a+b)}}\frac{1}{2^{2a+2b}} = O(1/\sqrt{a+b}).$$ □

This illustrates the advantage of balanced uniform vs. unbalanced crossover. However, this advantage is only relevant if the number of differing positions in x and y becomes large.

Exploiting the above statements, the expected optimization time of the (2+1) GA with Hamming-distance maximization and balanced uniform crossover improves to $O(n\log(n))$ for $n - B = \Theta(n)$.

Theorem 6.11. *For constant $0 < p_c < 1$, consider the* (2+1) GA *on* $\textsc{BoundMax}_B$ *with constraint $B = cn$ for constant c, using Hamming-distance maximization for tie-breaking. The expected optimization time is in $O(n\log(n))$.*

Proof. We split the proof in three stages. We first analyze the time until both individuals have exactly B 1-bits, then the time until both individuals have at most $\frac{\min(c,1-c)n}{2}$ many heavy 1-bits, and lastly the time until the optimal solution is created. In the following, we denote the two individuals of the GA with x and y.

For the first part, by employing Lemma 5.4, the expected number of iterations until both individuals have exactly B 1-bits is in $O(n \log(n))$. For the second part, we get from Lemma 6.1 that the time until the (1+1) EA creates an individual with at most $k := \frac{\min(c,1-c)n}{2}$ heavy 1-bits is $O(n)$. Since the number of heavy 0-bits of the best individual does not decrease if both individuals have B 1-bits, and any worsening by crossover only contributes a constant factor to the runtime, the expected time until the first individual has at most k heavy 1-bits is in $O(n)$. The expected time until this holds for both individuals is at most twice this.

For the third part of the proof, we employ the fitness level method [Wegener 2002] and define the fitness levels $A_{k,0}, A_{k,1}, A_{k,2}, A_{k,3}, A_{k-1,0}, A_{k-1,1}, A_{k-1,2}, A_{k-1,3} \ldots, A_{0,0}$ where the algorithm is in level $A_{i,j}$ for $j \in \{0,1,2,3\}$ if the individual with highest fitness has exactly i heavy 0-bits. Again, note that these fitness levels partition the set of all individuals with exactly B 1-bits. Each fitness level is partitioned in four sub levels with the following interpretations:

- $A_{i,0}$: Exactly one individual has i heavy 0-bits and the other has more than i heavy 0-bits.
- $A_{i,1}$: Both individuals have i heavy 0-bits and are duplicates.
- $A_{i,2}$: Both individuals have i heavy 0-bits and differ in at least one position in the first or second block but not in both blocks.
- $A_{i,3}$: Both individuals have i heavy 0-bits and differ in at least one position in the first and in the second block.

We analyze the probability $p_{i,j}$ of advancing from level $A_{i,j}$. Following the idea of the fitness-level method, the expected runtime is then upper bounded by $E[T] \leq \sum_{i=1}^{B} \sum_{j=0}^{3} \frac{1}{p_{i,j}}$.

Note that, in contrast to usual application of the fitness level method, it is possible to fall back from level $A_{i,3}$ to $A_{i,2}$. However, the probability that this happens before we transition to levels above $A_{i,3}$ is $o(1)$, so a simple restart argument shows that this has only a lower order impact on the runtime which we will ignore. Furthermore, no population can fall back from $A_{i,1}, A_{i,2}$, or $A_{i,3}$ to $A_{i,0}$ as this would require that the fitness of an individual decreases. No population can fall back from $A_{i,2}$ or $A_{i,3}$ to $A_{i,1}$ as this would imply a decrease of Hamming distance.

For probability $p_{i,0}$, we consider the event that the fittest individual gets duplicated by means of a standard mutation that does not flip any bit. This happens with a probability of at least $\frac{1-p_c}{2}(1 - \frac{1}{n})^n \geq \frac{1-p_c}{4e} = \Omega(1)$ which shows that $p_{i,0} = \Omega(1)$.

For $p_{i,1}$ and $p_{i,2}$, we note that in the levels $A_{i,1}$ and $A_{i,2}$, the first or the second block of x and y are identical. We consider the event of advancing to a following level by swapping a one and a zero within one of the identical blocks by means of the mutation operator. Since such a mutation does not change the fitness of the offspring but increases its Hamming distance to the other individuals by 2, it is always accepted. From the fact that there are $i \leq \min(c, 1-c)n/2$ heavy 0-bits and light 1-bits, respectively, in both x and y, we get that the probability of the described event is at least

$$\frac{i(\min(c,1-c)n - i)}{n^2}\left(1 - \frac{1}{n}\right)^{n-2} \geq \frac{i \min(c,1-c)n/2}{en^2}.$$

Hence, both $p_{i,1}$ and $p_{i,2}$ are in $\Omega(i/n)$. For $p_{i,3}$, we consider the event that crossover creates a solution with exactly B 1-bits and more than i heavy 1-bits. Since they differ in at least two positions in each block, we get from Lemma 6.9 that this probability is $\geq 1/6$.

We can thus estimate the expected runtime as

$$E[T] \leq \sum_{i=1}^{B} \sum_{j=0}^{3} \frac{1}{p_{i,j}} \leq \sum_{i=1}^{B} \left(\alpha + \beta \frac{n}{i} + \gamma\right),$$

where α, β, γ are positive constants. Hence, for some constant $\delta > 0$

$$E[T] \leq \sum_{i=1}^{B} \left(\alpha + \beta\frac{n}{i} + \gamma\right) \leq \delta n \log(B) = O(n \log(n)). \qquad \square$$

If B is only a constant away from n, we can show that none of the analyzed crossover scenarios bring any improvements over just a (1+1) EA with standard mutation. There is a constant probability of having a blocking light bit once the constraint is met, and the probability to remove it is in $O(1/n^2)$ for small d.

Theorem 6.12. *For the* (2+1) GA *using Hamming-distance maximization as well as for the island models employing balanced uniform or majority vote crossover, there is a constant d such that the expected optimization time when optimizing* $\textsc{BoundMax}_B$ *with constraint $B = n - d$ is in $\Theta(n^2)$.*

Proof. We choose $d = 1$. The probability of the only light bit in both initial individuals being a 1-bit is $1/4$, which results in a blocking light bit. In the proof for Proposition 5.2, we already showed that after an individual has reached $B = n - 1$ many 1-bits, the last bit is 1 with a probability of at least $1/2$. Consequently, the probability of having a blocking light bit after reaching B many 1-bits is at least $1/4$. For the island models, this is trivial. For majority vote, the probability is even higher as there are more bit strings available to initially share a blocking light bit. For Hamming-distance maximization, an application of the crossover can never remove a blocking bit on the way to B many 1-bits, and both individuals can be considered independently with respect to this probability.

No inheritance-respectful crossover operator can resolve such a blocking bit, so it needs to be removed by the standard mutation. For that, the only light bit needs to be swapped with the only 0-bit, resulting in a probability of at most $\frac{1}{n^2}$. This together with the constant probability of reaching this case yields the lower bound of $\Omega(n^2)$. Since standard mutation alone requires only $O(n^2)$ and optimization is finished when all individuals reach the optimum via standard mutation alone, that bound is also tight. $\square$

6.4 (2+1) Swap-GA

We have seen that the (2+1) GA achieves a runtime of $O(n \log(n))$ if $B = cn$ but takes $\Theta(n^2)$ steps for the case of $B = n - d$. On the other hand, the (1+1) Swap-EA takes time $\Theta(n^2)$ for $B = cn$ but $O(n \log(n))$ for $B = n - d$. In this section, we show that combining the two strategies together in the (2+1) Swap-GA yields an expected runtime of $O(n \log(n))$ in both cases, and, in fact, for all choices of B. The proof is given below in Theorem 6.14, it involves the following lemma.

Lemma 6.13. *Consider the* (1+1) Swap*-EA on* $\textsc{BoundMax}_B$ *with constraint B and starting on an individual with exactly B 1-bits. Let T be the random variable describing the number of steps until the algorithm finds a solution with at most $k \in \mathbb{N}^+$ heavy 0-bits. Then, $E[T]$ is in $O(\frac{nB-B^2}{k})$.*

Proof. Like in Lemma 6.1, we employ the fitness level method with fitness levels $A_{\leq k}, A_{k+1}, \ldots, A_B$, where A_i is the set of bit strings with i heavy 0-bits and $A_{\leq k}$ contains all bit strings with at most k such bits. The initial solution is at worst in A_B, and we are interested in the time until a solution in $A_{\leq k}$ is first sampled.

Let p_i be the probability of leaving the fitness level A_i in an iteration. We show that $p_i \geq \frac{(1-p_c)p_b \cdot i^2}{nB-B^2}$. To leave the level A_i, we only consider the event that swap mutation flips one of the i heavy 0-bits with one of the i light 1-bits. The probability for this event is $(1 - p_c)p_b \cdot \frac{i}{B} \cdot \frac{i}{n-B} \geq \frac{(1-p_c)p_b \cdot i^2}{nB-B^2}$.

We again apply the fitness level method. Let T' be the random variable denoting the number of iterations to reach level $A_{\leq k}$. We get

$$E[T'] \leq \sum_{i=k+1}^{B} \frac{1}{p_i} \leq \sum_{i=k+1}^{B} \frac{nB - B^2}{(1-p_c)p_b \cdot i^2} = \frac{nB - B^2}{(1-p_c)p_b} \sum_{i=k+1}^{B} \frac{1}{i^2}.$$

Just like in the proof for Lemma 6.1, the integral can be used to bound the sum. We get

$$E[T'] \leq \frac{nB - B^2}{(1-p_c)p_b} \left(-\frac{1}{B} + \frac{1}{k}\right) \leq \frac{nB - B^2}{(1-p_c)p_b \cdot k}.$$

This implies a total expected running time in $O(\frac{nB-B^2}{k})$. □

We proceed by proving the main result of this section and remark that the following proof has large intersections with the proof of our earlier and related statement Theorem 6.11.

Theorem 6.14. *For constant $0 < p_b, p_c < 1$, consider the* (2+1) Swap-GA *on* BoundMax$_B$*, using Hamming-distance maximization for tie-breaking. The expected optimization time is in $O(n \log(n))$.*

Proof. We split the proof in three stages. We first analyze the time until both individuals have exactly B 1-bits, then the time until both individuals have at most $\frac{\min(B,n-B)}{2}$ many heavy 1-bits, and lastly the time until the optimal solution is created. In the following, we denote the two individuals of the GA with x and y.

For the first part, by once more employing Lemma 5.4, the expected number of iterations until both individuals have exactly B 1-bits is in $O(n \log(n))$. For the second part, we get from Lemma 6.13 that the time until the (1+1) EA creates an individual with at most $k := \frac{\min(B,n-B)}{2}$ heavy 1-bits is in $O(n)$. Since the number of heavy 0-bits of the best individual does not decrease if both individuals have B 1-bits, and since the crossover and the standard mutations can only contribute a constant factor to the runtime, the expected time until the first individual has at most k heavy 1-bits is likewise in $O(n)$ for the GA. The expected time until this is the case for both individuals is at most twice as long.

For the third part of the proof, we employ the fitness level method [Wegener 2002] and define the fitness levels $A_{k,0}, A_{k,1}, A_{k,2}, A_{k,3}, A_{k-1,0}, A_{k-1,1}, A_{k-1,2}, A_{k-1,3} \ldots, A_{0,0}$ where the algorithm is in level $A_{i,j}$ for $j \in \{0,1,2,3\}$ if the individual with highest fitness has exactly i heavy 0-bits. Again, note that these fitness levels partition the set of all individuals with exactly B 1-bits. Each fitness level is partitioned in four sub levels with the following interpretations:

- $A_{i,0}$: Exactly one individual has i heavy 0-bits and the other has more than i heavy 0-bits.
- $A_{i,1}$: Both individuals have i heavy 0-bits and are duplicates.
- $A_{i,2}$: Both individuals have i heavy 0-bits and differ in at least one position in the first or second block but not in both blocks.
- $A_{i,3}$: Both individuals have i heavy 0-bits and differ in at least one position in the first and in the second block.

We analyze the probability $p_{i,j}$ of advancing from level $A_{i,j}$. Following the idea of the fitness-level method, the expected runtime is then upper bounded by

$$E[T] \leq \sum_{i=1}^{B} \sum_{j=0}^{3} \frac{1}{p_{i,j}}.$$

Note that, in contrast to usual application of the fitness level method, it is possible to fall back from level $A_{i,3}$ to $A_{i,2}$. However, the probability that this happens before we transition to levels

above $A_{i,3}$ is $o(1)$, so a simple restart argument shows that this has only a lower order impact on the runtime which we will ignore in the following.

For probability $p_{i,0}$, we consider the event that the fittest individual gets duplicated by means of a standard mutation that does not flip any bit. This happens with a probability of at least $\frac{(1-p_c)(1-p_b)}{2}(1-\frac{1}{n})^n \geq \frac{(1-p_c)(1-p_b)}{4e} = \Omega(1)$ which shows that $p_{i,0} = \Omega(1)$.

For $p_{i,1}$ and $p_{i,2}$, we note that in the levels $A_{i,1}$ and $A_{i,2}$, the first or the second block of x and y are identical. We consider the event of advancing to a following level by swapping a one and a zero within one of the identical blocks by means of the swap mutation operator. Since such a mutation does not change the fitness of the offspring but increases its Hamming distance to the other individuals by 2, it is always accepted. Recall that there are $i \leq \min(B, n-B)/2$ heavy 0-bits and light 1-bits, respectively, in both x and y. For the probability of swapping in the heavy block, we hence have

$$(1-p_c)p_b \frac{i}{n-B} \cdot \frac{B-i}{B} \geq (1-p_c)p_b \frac{i}{n-B} \cdot \frac{B}{2B} = \Omega\left(\frac{i}{n-B}\right).$$

For the probability of swapping in the light block, we analogously get

$$(1-p_c)p_b \frac{i}{B} \cdot \frac{n-B-i}{n-B} \geq (1-p_c)p_b \frac{i}{B} \cdot \frac{n-B}{2(n-B)} = \Omega\left(\frac{i}{B}\right).$$

Hence, both $p_{i,1}$ and $p_{i,2}$ are in $\Omega(i/n)$.

For $p_{i,3}$, we consider the event that crossover on both individuals creates a solution with exactly B 1-bits and more than i heavy 1-bits. Since both individuals differ in at least two positions in each block, we get from Lemma 6.9 that this probability is at least $1/6$.

We can thus estimate the expected runtime as

$$E[T] \leq \sum_{i=1}^{B} \sum_{j=0}^{3} \frac{1}{p_{i,j}} \leq \sum_{i=1}^{B} \left(\alpha + \beta\frac{n}{i} + \gamma\right),$$

where α, β, γ are constants greater than 0. This gives the existence of a constant $\delta > 0$ such that

$$E[T] \leq \sum_{i=1}^{B} \left(\alpha + \beta\frac{n}{i} + \gamma\right) \leq \delta n \log(B) = O(n \log(n)). \qquad \square$$

7 CONCLUSION

In this article, we gave runtime analyzes for several different settings of crossover on the test function $\textsc{BoundMax}_B$, a $\textsc{OneMax}$-like function with a constraint of at most B many 1s in a bit string. We showed how island-settings can be used to generate the optimum in one final crossover, and we contrasted this with settings where offspring from crossover can also serve as parents for crossover which, in our setting, leads to much better runtime guarantees.

Regarding the use of balanced crossovers, we strongly believe that any crossover should either (a) incorporate problem knowledge to gain performance or (b) be balanced, order unbiased, and inheritance-respectful. The reason for this is that, in the absence of meaning of the order of bits, the bit positions should be treated symmetrically. For this setting, we recommend the balanced uniform crossover, a balanced variant of the uniform crossover.

The formal analysis of crossover remains challenging. We see both potential and challenge in this area: Even though crossover has been the subject of study for decades, results remain somewhat scarce and further insights might be the key to understanding the success of evolutionary algorithms on real-world problems.

REFERENCES

Denis Antipov, Maxim Buzdalov, and Benjamin Doerr. 2020. Fast mutation in crossover-based algorithms. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 1268–1276. DOI: https://doi.org/10.1145/3377930.3390172

Jiah-Shing Chen and Jia-Leh Hou. 2006. A combination genetic algorithm with applications on portfolio optimization. In *Advances in Applied Artificial Intelligence*. Moonis Ali and Richard Dapoigny (Eds.), Lecture Notes in Computer Science, Springer, 197–206. DOI: https://doi.org/10.1007/11779568_23

Dogan Corus and Pietro S. Oliveto. 2018. Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms. *IEEE Transactions on Evolutionary Computation* 22, 5 (2018), 720–732. DOI: https://doi.org/10.1109/TEVC.2017.2745715

Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Martin S. Krejca, Per Kristian Lehre, Pietro S. Oliveto, Dirk Sudholt, and Andrew M. Sutton. 2016. Escaping local optima with diversity mechanisms and crossover. In *Proceedings of the 2016 Genetic and Evolutionary Computation Conference*. 645–652. DOI: https://doi.org/10.1145/2908812.2908956

Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Martin S. Krejca, Per Kristian Lehre, Pietro S. Oliveto, Dirk Sudholt, and Andrew M. Sutton. 2018. Escaping local optimausing crossover with emergent diversity. *IEEE Transaction on Evolutionary Computation* 22, 3 (2018), 484–497. DOI: https://doi.org/10.1109/TEVC.2017.2724201

Benjamin Doerr and Carola Doerr. 2015. A tight runtime analysis of the $(1 + (\lambda, \lambda))$ genetic algorithm on OneMax. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, New York, NY, 1423–1430. DOI: https://doi.org/10.1145/2739480.2754683

Benjamin Doerr, Philipp Fischbeck, Clemens Frahnow, Tobias Friedrich, Timo Kötzing, and Martin Schirneck. 2019. Island models meet rumor spreading. *Algorithmica* 81, 2 (2019), 886–915. DOI: https://doi.org/10.1007/s00453-018-0445-2

Benjamin Doerr and Timo Kötzing. 2021. Lower bounds from fitness levels made easy. In *Proceedings of the 2021 Genetic and Evolutionary Computation Conference*. ACM, 1142–1150. DOI: https://doi.org/10.1145/3449639.3459352

Tobias Friedrich, Timo Kötzing, Martin S. Krejca, Samadhi Nallaperuma, Frank Neumann, and Martin Schirneck. 2016. Fast building block assembly by majority vote crossover. In *Proceedings of the 2016 Genetic and Evolutionary Computation Conference*. 661–668. DOI: https://doi.org/10.1145/2908812.2908884

Tobias Friedrich, Timo Kötzing, J. A. Gregor Lagodzinski, Frank Neumann, and Martin Schirneck. 2020. Analysis of the (1+1) EA on subclasses of linear functions under uniform and linear constraints. *Theoretical Computer Science* 832 (2020), 3–19. DOI: https://doi.org/10.1016/j.tcs.2018.04.051

Tobias Friedrich, Timo Kötzing, Aishwarya Radhakrishnan, Leon Schiller, Martin Schirneck, Georg Tennigkeit, and Simon Wietheger. 2022. Crossover for cardinality constrained optimization. In *Proceedings of the 2022 Genetic and Evolutionary Computation Conference*. 1399–1407. DOI: https://doi.org/10.1145/3512290.3528713

Thomas Jansen. 2015. On the black-box complexity of example functions: The real jump function. In *Proceedings of the 13th Conference on Foundations of Genetic Algorithms*. 16–24. DOI: https://doi.org/10.1145/2725494.2725507

Thomas Jansen and Dirk Sudholt. 2010. Analysis of an asymmetric mutation operator. *Evolutionary Computation* 18, 1 (Mar 2010), 1–26. DOI: https://doi.org/10.1162/evco.2010.18.1.18101

Thomas Jansen and Ingo Wegener. 2002. The analysis of evolutionary algorithms—A proof that crossover really can help. *Algorithmica* 34, 1 (2002), 47–66. DOI: https://doi.org/10.1007/s00453-002-0940-2

Svante Janson. 2018. Tail bounds for sums of geometric and exponential variables. *Statistics & Probability Letters* 135 (Apr 2018), 1–6. DOI: https://doi.org/10.1016/j.spl.2017.11.017

Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. 2021. A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications* 80, 5 (2021), 8091–8126. DOI: https://doi.org/10.1007/s11042-020-10139-6

Timo Kötzing, Dirk Sudholt, and Madeleine Theile. 2011. How crossover helps in pseudo-boolean optimization. In *Proceedings of the 2011 Genetic and Evolutionary Computation Conference*. 989–996. DOI: https://doi.org/10.1145/2001576.2001711

Per Kristian Lehre and Carsten Witt. 2012. Black-box search by unbiased variation. *Algorithmica* 64, 4 (2012), 623–642. DOI: https://doi.org/10.1007/s00453-012-9616-8

Luca Manzoni, Luca Mariot, and Eva Tuba. 2020. Balanced crossover operators in genetic algorithms. *Swarm and Evolutionary Computation* 54 (2020), 100646. DOI: https://doi.org/10.1016/j.swevo.2020.100646

Thorsten Meinl and Michael R. Berthold. 2009. Crossover operators for multiobjective k-subset selection. In *Proceedings of the 2009 Genetic and Evolutionary Computation Conference*. 1809–1810. DOI: https://doi.org/10.1145/1569901.1570173

Alberto Moraglio and Riccardo Poli. 2004. Topological interpretation of crossover. In *Proceedings of the 2004 Genetic and Evolutionary Computation Conference*. Kalyanmoy Deb (Ed.), Springer, 1377–1388. DOI: https://doi.org/10.1007/978-3-540-24854-5_131

Frank Neumann, Pietro Simone Oliveto, Günter Rudolph, and Dirk Sudholt. 2011. On the effectiveness of crossover for migration in parallel evolutionary algorithms. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*. ACM, 1587–1594. DOI: https://doi.org/10.1145/2001576.2001790

Frank Neumann and Ingo Wegener. 2007. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science* 378, 1 (Jun 2007), 32–40. DOI: https://doi.org/10.1016/j.tcs.2006.11.002

Nicholas J. Radcliffe. 1994. The algebra of genetic algorithms. *Annals of Mathematics and Artificial Intelligence* 10, 4 (Dec 1994), 339–384. DOI: https://doi.org/10.1007/BF01531276

Jonathan E. Rowe and Michael D. Vose. 2011. Unbiased black box search algorithms. In *Proceedings of the 2011 Genetic and Evolutionary Computation Conference*. 2035–2042. DOI: https://doi.org/10.1145/2001576.2001850

Dirk Sudholt. 2017. How crossover speeds up building block assembly in genetic algorithms. *Evolutionary Computation* 25, 2 (2017), 237–274. DOI: https://doi.org/10.1162/EVCO_a_00171

Richard A. Watson and Thomas Jansen. 2007. A building-block royal road where crossover is provably essential. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*. ACM, 1452–1459. DOI: https://doi.org/10.1145/1276958.1277224

Ingo Wegener. 2002. Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In *Evolutionary Optimization*. Ruhul Sarker, Masoud Mohammadian, and Xin Yao (Eds.), Springer, 349–369. DOI: https://doi.org/10.1007/0-306-48041-7_14

L. Darrell Whitley, Francisco Chicano, and Brian W. Goldman. 2016. Gray box optimization for Mk landscapes (NK landscapes and MAX-kSAT). *Evolutionary Computation* 24, 3 (2016), 491–519. DOI: https://doi.org/10.1162/EVCO_a_00184

Carsten Witt. 2014. Fitness levels with tail bounds for the analysis of randomized search heuristics. *Information Processing Letters* 114, 1 (Jan 2014), 38–41. DOI: https://doi.org/10.1016/j.ipl.2013.09.013

Received 7 October 2022; revised 18 May 2023; accepted 21 May 2023