Greedy versus Curious Parent Selection for Multi-Objective Evolutionary Algorithms

Denis Antipov[1] [0000−0001−7906−096X], Timo Kötzing[2] [0000−0002−1028−5228], and Aishwarya Radhakrishnan[2] [0000−0002−5667−8780]

[1] University of Adelaide, Adelaide, Australia
denis.antipov@adelaide.edu.au
[2] Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
{timo.koetzing,aishwarya.radhakrishnan}@hpi.de

Abstract. From the literature we know that simple evolutionary multi-objective algorithms can optimize the classic two-objective test functions OneMinMax and CountingOnesCountingZeroes in $O(n^2 \log n)$ expected time. We extend this result to any pair of generalized OneMax functions and show that, if the optima of the two functions are d apart, then (G)SEMO has an expected optimization time of $O(dn \log(n))$.
In an attempt to achieve better optimization times, some algorithms consider parent selection. We show that parent selection based on the curiosity-based novelty search can improve the optimization time to $O(n^2)$ on OneMinMax. By contrast, we show that greedy parent selection schemes can be trapped with an incomplete Pareto front for superpolynomial time.
Finally, we provide experimental results on the two-objective optimization of linear functions.

Keywords: Evolutionary Algorithm · Multi-Objective Optimization · Run time analysis.

1 Introduction

While evolutionary algorithms [16] might be most famous for applications on single-objective problems, the setting of optimizing multiple criteria at once is particularly suitable for an approach with population-based methods, since different candidate solutions might be incomparable: while one solution is better than another in terms of the first criterion, the situation is reversed in terms of the second criterion, and so on. Thus it makes sense to retain all non-dominated solutions (where no other solution is better in *all* objectives), naturally giving a population of solutions. The analysis of the search behavior and search performance has been the subject of significant theoretical analysis [10,11,12,17,23,28].

Core starting point of all theoretical research is the classic benchmark problem OneMax, which was extended to the setting of two objectives in two ways. The first uses the classic OneMax function as one objective and the direct opposite, *minimizing* the number of 1s instead of maximizing it, as the other objective.

Using these two objectives is called OneMinMax. The second extension considers less conflicting bits: While the second half of the bits stay in conflict, the first half are shared. This is called CountingOnesCountingZeroes (COCZ).

The behavior of the classic SEMO (Simple Evolutionary Multi-Objective) and GSEMO (Global Simple Evolutionary Multi-Objective) algorithms is well-understood on these two problems (see [20,22,4,26]). The expected run time of SEMO and GSEMO to cover the whole Pareto front while minimizing OneMinMax and COCZ is $\Theta(n^2 \log(n))$ and the theoretical analyses can be found in [20,22,4].

For the case of single-objective optimization, the broader class of linear functions [14] gives an important extension of the simple OneMax function, extending it to a sizable class of functions. This was a driver for further development of the field [27]. While OneMax as a member of this class has been analyzed for the two-objective case, no other linear functions where considered.

With this paper, we first provide a general definition of two-objective problems where both objectives are derived from the OneMax test function; we call this OMC, the OneMax function class. Analogously, we define LFC, the linear function class. In Section 4 we introduce and discuss these function classes and study some of their properties. We also include a proof of the expected run time of (G)SEMO on two elements of OMC being $O(dn \log(n))$ in dependence on the distance $d > 0$ of the optima of the two objective functions. Note that this shows the smooth transition of run time $O(n \log(n))$ when using twice the same objective and $O(n^2 \log(n))$ for complementary objectives (as in OneMinMax) and also recovers the run time bound for COCZ.

The considered algorithms typically waste a lot of time reconsidering old search points which are already optimal and where no more progress can be made in the proximity. This inspires algorithms based on considering new search points rather than reconsidering old ones. This paradigm is called novelty search and in the literature the algorithm is known as fair evolutionary multi-objective optimizer (FEMO) [18,22,23]. In Section 6 we consider a simple variant of such an algorithm which maintains, for each phenotype, a counter of how often it was considered for creating offspring. Each iteration, an individual with minimal counter is considered for creating offspring. We show, in Theorem 4, that this algorithm has an expected optimization time of $O(n^2)$ on OneMinMax, improving over (G)SEMO.

Novelty search modifies which individuals are considered for creating offspring, while leaving the rest of the algorithm as is. This is called a *parent selection scheme* and the literature knows a variety of other mechanisms [4,5]. These schemes typically rank all individuals of the population according to how promising they are to create relevant offspring and then prefer more promising ones over less promising ones. In Section 7 we show that, for many ranking schemes and too radically greedy preference of more promising points, we get super-polynomial optimization on OneMinMax with constant probability (see Theorem 6). We consider this as a cautionary tale that parent selection schemes need to reconsider less promising search points from time to time, even on very easy fitness landscapes (such as OneMinMax).

Finally, in Section 8, we provide experimental evidence for the expected run time performance of GSEMO on two anti-aligned LFC functions without any shared bits and GSEMO on two anti-aligned LFC functions without any conflicted bits. Our results hint at an asymptotic run time of $O(n^2 \cdot \log(n))$ for anti-aligned LFC functions without any shared bits and $O(n \cdot \log(n))$ for anti-aligned LFC functions without any conflicted bits.

The remainder of this paper first gives some discussion on further related work (see Section 2). We give important definitions in Section 3. We introduce OMC formally (along with the extension to linear functions) in Section 4 and analyze the run time of (G)SEMO on OMC in Section 5. We analyze novelty search in Section 6 and greedy parent selection in Section 7. We conclude with some experiments in Section 8. Many proofs are not included into this document, but can be found in the supplementary material [1].

2 Related Works

In [23], a first run time analysis was conducted on the simple multi-objective optimization algorithm (SEMO) on minimizing LeadingOnesTrailingZeros (LOTZ). This work was extended in [22] to the fair and the greedy multi-objective optimization algorithms (FEMO, GEMO) and the multi-start (1+1) EA on COCZ and LOTZ.

The global simple multi-objective optimization algorithm (GSEMO) was analyzed on LOTZ in [19] along with a lower bound on GSEMO for a general class of pseudo-boolean functions. GSEMO and GSEMO with asymmetric mutation operator on plateau functions and set cover instances were studied in [3,17]. In [18], the performance of GSEMO and Global-FEMO algorithms on plateaus, plateaus with gap and dual path were analyzed. In [25], analysis of GSEMO with mixed strategy (mixing selection mechanisms) on ZPLG (ZeroMax, a plateau, and a path with little gaps) and SPG (shortest path and gaps) can be found.

The algorithms SEMO and GSEMO with crossover operators on COCZ and minimum spanning tree (MST) problems were studied in [26]. The first analysis of SEMO optimizing OneMinMax was given in [20]. The OneMinMax function was again analyzed in [8], but using the $(\mu + 1)$-SIBEA$_D$ algorithm. The decomposition-based multi-objective evolutionary algorithms (MOEAs) were introduced in [24] and analyzed on COCZ and LOTZ.

A diversity-based parent selection mechanism (based on hypervolume contribution) for SEMO and GSEMO was given in [4,5] and studied on minimizing OneMinMax and LOTZ. In [12], SEMO and GSEMO were studied on optimizing the OneJumpZeroJump function. An offspring selection mechanism which uses the total Hamming distance as a diversity measure was given in [2] and the OneMinMax function was again analyzed in this setting.

3 Preliminaries

In this section we give some definitions, the algorithms we analyze and some notations which we use throughout the paper. We use the following theorem in some of our proofs.

Theorem 1 (Multiplicative Drift Theorem [21]) *Let $(X_t)_{t \in \mathbb{N}}$ be a random process over $\mathbb{R}$, $x_{\min} > 0$, $\delta > 0$ and let $T = \min\{t \mid X_t < x_{\min}\}$. Furthermore, suppose that*

1. *$X_0 \geq x_{\min}$ and, for all $t \leq T$, it holds that $X_t \geq 0$, and that*
2. *for all $t < T$, we have $X_t - E[X_{t+1} \mid X_0, \ldots, X_t] \geq \delta X_t$.*

Then

$$E[T \mid X_0] \leq \frac{1 + \ln\left(\frac{X_0}{x_{\min}}\right)}{\delta}.$$

We analyze the simple multi-objective optimizer (SEMO) and the global multi-objective optimizer (GSEMO) algorithms (see Algorithm 1) on different bi-objective functions in this paper. The only difference between SEMO and GSEMO is the mutation step. In SEMO, at the mutation step, a bit position is chosen uniformly at random and flipped (one bit mutation). In GSEMO, each bit position is flipped with probability $1/n$ (standard bit mutation).

The initial population has only one individual chosen at random from $\{0,1\}^n$. An individual x dominates another individual y ($x \succeq y$) if and only if $f_1(x) \leq f_1(y)$ and $f_2(x) \leq f_2(y)$. Note that we use slightly different Pareto dominance relation which prefers the offspring if the offspring has the same fitness as any of the other individuals existing in the population. We use the term genotype to refer to the individuals in the input domain and the term phenotype to refer to the fitness vector.

Algorithm 1: (Global) Simple Evolutionary Multi-objective Optimizer ((G)SEMO) minimizing $f = (f_1, f_2)$.

$x \leftarrow$ choose u.a.r from $\{0,1\}^n$, $P \leftarrow \{x\}$;
while *termination criteria not met* **do**
  select parent x from P u.a.r;
  $x' \leftarrow \text{mutate}(x)$;
  $P \leftarrow P \setminus \{z \in P \mid x' \succeq z\}$;
  if $\nexists z \in P$ ***s.t*** $(z \succeq x')$ **then** $P \leftarrow P \cup \{x'\}$;

When discussing greedy parent selection schemes, we use the following (simplified) definition of the hypervolume contribution for 2 objectives, usually used in minimization problems.

Definition 1. *Consider a bi-objective function $f = (f_1, f_2)$ and a population of points $P = (x_1, \dots, x_\mu)$ which do not dominate each other (in terms of f) and are sorted in the ascending order of their f_1 value. Let $r = (r_1, r_2)$ be a reference point such that $r_1 \geq f_1(x_\mu)$ and $r_2 \geq f_2(x_1)$. For all $i \in [1..\mu]$ let $a_i = f_1(x_i)$ and let $b_i = f_2(x_i)$. Let also $a_{\mu+1} = r_1$ and $b_0 = r_2$. Then for all $i \in [1..\mu]$ the hypervolume contribution (HVC) of point $x_i \in P$ is $(a_{i+1} - a_i) \cdot (b_{i-1} - b_i)$.*

4 Linear Multi-Objective Functions

In this section we analyze two classes of functions: first, the OneMax function class, where each fitness function measures the Hamming distance to some optimal bit string. Second, the linear function class, where each bit has a weight and fitness is the sum of the weights of incorrect bits.

Formally, for each $a \in \{0,1\}^n$, we let

$$\mathrm{OM}_a : \{0,1\}^n \to \mathbb{R}, x \mapsto H(a, x),$$

where H is the Hamming distance between two bit strings. We define the OneMax class as

$$\mathrm{OMC} = \{\mathrm{OM}_a \mid a \in \{0,1\}^n\}.$$

Note that the OneMax class has been studied before in the context of black-box optimization [7,13,15]. The most famous example from OMC is OM := OM_{1^n} which is minimal at 1^n. For two-objective optimization, we also care for the exact opposite, ZeroMax, denoted as ZM := OM_{0^n}.

Similarly, we can define the linear function class LFC as follows. For each $w \in \mathbb{R}^{n+1}$, we let

$$f_w : \{0,1\}^n \to \mathbb{R}, x \mapsto w_{n+1} + \sum_{i=1}^{n} w_i\, x_i.$$

Note that we use the constant w_{n+1} (a) as an offset, so that all function values are non-negative and can be more nicely depicted in a diagram; and (b) so that it is formally true that each OneMax function is a linear function, which they intuitively are (for example, for $a = 1^n$ we need $\forall i \in [1..n] : w_i = -1$ and $w_{n+1} = n$).

In the literature we frequently find the additional restrictions $w_{n+1} = 0$ and $\forall i \leq n : w_i > 0$; or even $\forall i < n : w_i > w_{i+1} > 0$. These can be assumed without loss of generality to simplify the exposition or the proof in the context of single objective optimization. However, in the context of optimizing two such functions simultaneously, and with the algorithm potentially making decisions not just based on the ranking of search points (but, for example, also based on hypervolume covered), we prefer this more general definition here.

We define the linear function class as

$$\mathrm{LFC} = \left\{f_w \mid w \in \mathbb{R}^{n+1}\right\}.$$

We have the following theorem about the two function classes and the proof can be found in the supplementary material [1].

Theorem 2 *We have* OMC $\subseteq$ LFC*, and both* OMC *and* LFC *are closed under isomorphisms of the hypercube.*

We use the following definition to talk about the fitness landscape of two linear fitness functions.

Definition 2. *Let two linear functions f_w, f_v be given. We call the set $I = \{i \in [n] \mid w_i \cdot v_i \geq 0\}$ the* shared bits*, since there is a bit setting which is optimal for both f_w and f_v. We call $[n] \setminus I$ the* conflicted bits*. We call the number of conflicted bits the* Pareto dimension*, since all elements on the Pareto front agree on the shared bits and only differ on conflicted bits (discarding the case of weights of* 0*). We frequently denote the Pareto dimension by d.*

If all n bits are conflicted, then we call f_w and f_v complementary *(since their unique global optima are complementary).*

We call f_w and f_v anti-aligned *if ordering the bits descendingly according to $|w_i|$-value leads to an an ascending ordering according to $|v_i|$-value. In other words: the more significant bit positions of w are, the less significant bit positions of v (and vice versa).*

5 SEMO and GSEMO on OMC

Here we give a generalization of the OneMinMax and COCZ analysis to the situation where the optima can share any number of bits (rather than either 0 bits as for OneMinMax or $n/2$ bits as in COCZ). Let $\log^+(x) = \max\{\log(x), 1\}$.

Theorem 3 *Let $a, b \in \{0,1\}^n$, $a \neq b$, let $d = d_H(a, b)$ and $0 < d < n$. Then (G)SEMO minimizing* $(\mathrm{OM}_a, \mathrm{OM}_b)$ *takes $O(dn\log(n))$ function evaluations in expectation to discover the full Pareto front of size $d+1$.*

Proof. First we show that the expected time for (G)SEMO to find an individual on the Pareto front is $O(dn\log^+(n-d))$ using the multiplicative drift theorem (see Theorem 1).

Let T_1 be the time taken by (G)SEMO to find an individual on the Pareto front, and let I be the set of all shared bits, i.e., $I = \{i \in [n] \mid a_i = b_i\}$. Since $d = d_H(a, b)$, $|I| = n - d$. Also, an individual x is on the Pareto front if and only if all shared bits of a and b (elements of I) are set correctly, i.e., $\sum_{i\in I} |a_i - x_i| = \sum_{i\in I} |b_i - x_i| = 0$.

For any $t > 0$, let P^t be the parent population at iteration t. For any $t < T_1$, let $X^t = \arg\min_{x\in P^t}\{\sum_{i\in I} |a_i - x_i|\}$. Then we claim that, $X^t \geq X^{t+1}$, i.e., an individual with less correct shared bits will not dominate an individual with more correct shared bits. If an individual x has $1 < i \leq n$ more shared bits set correctly than another individual y, then for y to dominate x the individual y should have i more conflicted (non-shared) bits (than x) set correctly with respect to a and i more conflicted bits set correctly with respect to b. This is not possible, since setting a conflicted bit correctly with respect to a implies that this conflicted bit is set incorrectly with respect to b. Therefore, for all t we have, $X^t \geq X^{t+1}$.

We claim that at any time t, the population P^t has at most $d+1$ individuals. Since there are only d conflicted bits, if there are $d+2$ individuals in the population then, by the pigeonhole principle, there is an $i \in [0..d]$ such that there are two distinct individuals in the population in which i conflicted bits are set correctly with respect to a. Since both the individuals exist in the population, their fitness is different. Therefore, one of them has more shared bits set correctly than the other, which implies that one individual dominates the other. This contradicts the definition of (G)SEMO since it only stores non-dominated individuals in its population.

Now we claim that $\Pr\big(X^t - X^{t+1} = 1 \mid X^t\big) \geq \frac{X^t}{e(d+1)n}$. As the maximum size of the population is $d+1$ and the mutation operator can choose an individual contributing to the potential X^t and flip exactly one of the X^t positions where $a_i \neq x^t$ with probability $\frac{1}{n}$ in the case of SEMO and with probability at least $\frac{1}{en}$ in the case of GSEMO.

By the multiplicative drift theorem (Theorem 1) and since we have $X^0 \leq n-d$, the expected time taken by (G)SEMO to find an individual on the Pareto front $E[T_1]$ is $O(dn \log^+(n-d))$.

Next, we show that the expected time to cover the Pareto front after the algorithm finds an individual on the Pareto front is $O(dn \ln(d))$. At time T_1, when the algorithm finds an individual x^T on the Pareto front for the first time, the fitness of this first individual cannot be more than d in both objectives, since, as we mentioned before, an individual is on the Pareto front if and only if all shared bits of a and b are set correctly. Let the fitness of this first individual be $(i, d-i)$, where $0 \leq i \leq d$.

Let $Y_t = -1$, if all fitness vectors from $(0, d)$ to $(i, d-i)$ are in population P^t, and let it be the maximum $j < i$ such that we do not have fitness $(j, d-j)$ in population P^t otherwise. Similarly, let Z_t be $d+1$, if we have all fitness vectors from $(i, d-i)$ to $(d, 0)$ in the population P^t, and let it be the minimum $k > i$ such that $(k, d-k)$ is not in P^t otherwise. Then the Pareto front is covered, iff $Y_t = -1$ and $Z_t = d+1$. Consider first time T_2' until Y_t reaches -1. If $Y_t = j$, then to decrease it we can choose an individual with fitness $(j+1, d-j-1)$ (which exists in the population) with probability at least $\frac{1}{d+1}$ and flip exactly one one-bit in it with probability $\frac{j+1}{n}$ or $\frac{j+1}{en}$ for SEMO or GSEMO respectively. Since for each value of j we decrease Y_t only once, the total expected time until we have $Y_t = -1$ is at most $E[T_2'] \leq \sum_{j=0}^{i-1} \frac{en(d+1)}{j+1} = O(nd \log^+(i))$. Similarly, we can show that the expected time T_2'' until Z_t reaches $d+1$ is at most $O(nd \log^+(d-i))$. We then have that $T_2 \leq T_2' + T_2''$ and therefore, it is at most $O(nd \log(d))$.

Overall, (G)SEMO takes $O(dn \log^+(n-d) + dn \log(d)) = O(dn \log(n))$ iterations in expectation while minimizing $(\mathrm{OM}_a, \mathrm{OM}_b)$ to discover the full Pareto front of size $d+1$. $\square$

6 Novelty Search

Consider as an order scheme the ranking of all individuals by how often they have been considered for creating offspring since entering the population, from least to most frequently considered. In a sense, we want to explore under-explored parts of the search space, and we want to find novel areas. This can lead to speed-ups in exploration of the Pareto front, as the next theorem shows.

We distinguish two cases for novelty search: resetting the offspring counter when an individual was replaced by one with the exact same phenotype, and not doing such a reset. That is, when we reset the counter, we actually counting the number of times the *genotype* has been selected as a parent, and when we do not reset the counter, we count the number of such times for the *phenotype*. The following theorem shows a speed-up of the latter approach on OneMinMax compared to the standard parent selection. We believe that resetting the counter when replacing individuals with the same phenotype leads to a behavior much like uniform parent selection, which is not interesting for us.

Theorem 4 *(G)SEMO paired with the the novelty ranking without resets which always chooses a parent that has been considered the smallest number of times finds the whole Pareto-front on* OneMinMax *in $O(n^2)$ expected iterations.*

Proof. All individuals are on the Pareto-front, since more 1s contribute positively to minimize the objective ZM and negatively to the objective OM and vice versa in the case of more 0s. Any two individuals $x, y \in \{0,1\}^n$ either have the same number of 1s, which leads to the same fitness, or one of them has more 1s than the other, which implies that neither x dominates y nor y dominates x. Therefore, the set $\{(i, n-i) \mid 0 \leq i \leq n\}$ is the set of all possible fitness values which corresponds to $n+1$ individuals on the Pareto-front.

For reasons of space, in the rest of this proof we consider SEMO. The proof for GSEMO uses the same arguments, but has slightly different constants.

We break down the total run time into time taken for the following events to happen. For any $0 \leq i \leq n-1$, let X_i be the random variable which denotes the time taken to find an individual x with fitness $(i+1, n-i-1)$ after the algorithm has found an individual with fitness $(i, n-i)$ and the algorithm has chosen this individual at least n times for offspring creation. For any $1 \leq i \leq n$, let Y_i be the random variable which denotes the time taken to find an individual y with fitness $(i-1, n-i+1)$ after the algorithm has found an individual with fitness $(i, n-i)$ and has chosen this individual at least n times for offspring creation. At any given iteration, when individual with fitness $(i, n-i)$ is chosen as a parent, this individual has either not yet been selected for offspring selection n times or the algorithm tries to find a new individual in the Pareto front which is not in the population by mutation. Therefore, the expected time T to find all the elements on the Pareto-front is

$$E[T] \leq \sum_{i=0}^{n-1}(E[X_i]+n) + \sum_{i=1}^{n}(E[Y_i]+n) \leq \sum_{i=0}^{n-1} E[X_i] + \sum_{i=1}^{n} E[Y_i] + 2n^2. \quad (1)$$

Now we calculate upper bounds on the expectation of the random variables X_i and Y_i for a given i. The probability that an individual with fitness $(i, n-i)$ does not lead to an offspring with fitness $(i+1, n-i-1)$ after being chosen n times is $(1-\frac{n-i}{n})^n \leq e^{i-n}$ and the probability that an individual with fitness $(i, n-i)$ does not lead to an offspring with fitness $(i-1, n-i+1)$ after being chosen n times is $(1-\frac{i}{n})^n \leq e^{-i}$. The number of function evaluations needed for an individual with fitness $(i, n-i)$ to produce an offspring with fitness $(i+1, n-i-1)$ by flipping exactly one bit follows the $\mathrm{Geo}(\frac{n-i}{n})$ geometric distribution and each failure costs at most n function evaluations since every other individual in the population must be selected at least as many times as this desired individual for offspring selection before this individual can be selected again. We note that if a new fitness appears in the population at this stage, then we can wait for more than n iterations before we chose our individual with fitness $(i, n-i)$ again, however those iterations when we choose an individual with this new fitness do not go to the cost of our mistake, but to the n iterations which are allocated for that new fitness in the corresponding term in eq. (1) or they go to the price of our previous mistakes. Similarly, the number of function evaluations needed for an individual with fitness $(i, n-i)$ to produce an offspring with fitness $(i-1, n-i+1)$ by flipping exactly one bit follows the $\mathrm{Geo}(\frac{i}{n})$ with the cost of at most n function evaluations for each failure. Therefore, for $0 \leq i \leq n-1$, $X_i \leq ne^{i-n} \cdot \mathrm{Geo}(\frac{n-i}{n})$ and for $1 \leq i \leq n$, $Y_i \leq ne^{-i} \cdot \mathrm{Geo}(\frac{i}{n})$. Thus, from eq. (1) we have

$$\begin{aligned} E[T] &\leq \sum_{i=0}^{n-1} E\left[ne^{i-n} \cdot \mathrm{Geo}\left(\frac{n-i}{n}\right)\right] + \sum_{i=1}^{n} E\left[ne^{-i} \cdot \mathrm{Geo}\left(\frac{i}{n}\right)\right] + 2n^2 \\ &= n^2 \sum_{i=0}^{n-1} \frac{1}{(n-i)(e^{n-i})} + n^2 \sum_{i=1}^{n} \frac{1}{ie^i} + 2n^2 = O(n^2). \end{aligned}$$

□

We have the following corollary on strictly monotone increasing functions and the proof can be found in the supplementary material [1].

Corollary 5 *Let $f, g\colon \mathbb{R} \to \mathbb{R}$ be strictly monotone increasing. Then the novelty ranking paired without reset which always chooses an individual that has been considered least number of times (top individual) leads to a run time of $O(n^2)$ on minimizing $(f(OM), g(ZM))$.*

7 Counter-example for Phenotype-based Methods

In this section we show that the parent selection methods which are based on the phenotype of the points in the population might be decisive even on very simple problems. We consider GSEMO with an exaggerated greedy phenotype-based parent selection: it always chooses one of the two points with the largest

HVC (Definition 1) as a parent, each with probability $\frac{1}{2}$. We call this algorithm GSEMO_2 for brevity.

We study this algorithm on OneMinMax with a reference point $(2n, 2n)$, so that the largest HVC is always yielded by the two *edge* points in the population (the points with the largest and the smallest numbers of one-bits), and therefore one of them is always chosen as a parent. The main result of this section is the following theorem, which demonstrates an ineffectiveness of the GSEMO_2.

Theorem 6 *With probability $\Omega(1)$ the $GSEMO_2$ optimizing* OneMinMax *with reference point $(2n, 2n)$ does not find all points in the Pareto front in polynomial time.*

We split the proof of Theorem 6 into three stages. The first stage of the proof shows that a run of the algorithm with high probability occurs in a particular initial state, where the two edge points are in linear distance from each other (in phenotype space), but they are still not too far away from the initial search point. In the second stage we show that starting from the initial state, we are very likely to create a *hole* in the population, when we get an edge point in distance at least two (again, in the phenotype space) from the nearest other point in the population. In the last stage we show that once we get a hole, with constant probability it stays in the population for a super-polynomial time. For reasons of space, we omit the analysis of the first two stages, but it can be found in supplementary material [1].

Theorem 6 resembles Theorem 8.1 in [5], where a similar result was proven for the GSEMO with a similar (but artificially modified) greedy parent selection on LOTZ. The main difference of our result is that we use a much more simple function, for which all points in the search space are Pareto optimal, thus we do not need to modify the selection mechanism as in [5]. Another significant difference is that in the third stage of our proof extending the front is less likely than covering the hole, while for LOTZ these events are equally likely. Despite this, the hole is also likely to stay on OneMinMax.

We use the following notation. By x_t we denote the individual in the population with the maximum OneMax value after iteration t, and by y_t the one with the minimum OneMax value. Note that in iteration $t+1$ we always choose as a parent either x_t or y_t, since they are the edge points. In our proofs we also use an arbitrary small constant ε, which can be any value in $(0, \frac{1}{10})$. For simplicity we also assume that n is even and εn is an integer. We start the proof with several auxiliary results.

Lemma 7 *Let ω_t, $t \in \mathbb{N}$, be a sequence of random experiments. Let also A_t and B_t be sequences of events over the corresponding probabilistic spaces. Let C_t be another sequence of events such that $C_t = \cap_{j=1}^{t-1} \overline{A_j}$ (that is, C_t is the event that A_j did not occur before time t) and let τ be the first time when A_t occurs, that is, $\tau = \min\{t \mid \omega_t \in A_t\}$ and assume that $\Pr[\tau = +\infty] = 0$.*

(a) If there exists p such that, for all $t \in \mathbb{N}$, we have $\Pr\left[B_t \mid A_t \cap C_t\right] \le p$, then $\Pr[B_\tau] \le p$.

(b) If there exists q such that, for all $t \in \mathbb{N}$, we have $\Pr[B_t \mid A_t \cap C_t] \ge q$*, then* $\Pr[B_\tau] \ge q$*.*

Proof. We prove only (a), since the proof of (b) is analogous. Event $\tau = t$ occurs, iff A_t occurs and all A_j for $j \in [1..t-1]$ do not occur, that is, it is equal to event

$$A_t \cap \left(\bigcap_{j=1}^{t-1} \overline{A_j}\right) = A_t \cap C_t,$$

hence by condition we have $\Pr[B_t \mid \tau = t] \le p$.

Since $\Pr[\tau = +\infty] = 0$, we can use the law of total probability.

$$\Pr[B_\tau] = \sum_{t=1}^{+\infty} \Pr[\tau = t] \Pr[B_t \mid \tau = t] \le \sum_{t=1}^{+\infty} \Pr[\tau = t] \cdot p = p. \qquad \square$$

We now show that if we create a hole in our population, which is not too far from, but also not too close to the center of the Pareto front, then with at least a constant probability we move our edge points in a linear distance from this hole before we fill it.

Lemma 8 *Consider a run of the GSEMO$_2$ on* OneMinMax*. Assume that at some iteration t_0 we have some $i \in [\frac{n}{2} + 2\varepsilon n..\frac{n}{2} + 4\varepsilon n]$ such that*

(1) we do not have fitness $(i, n-i)$ in population,
(2) $\mathrm{OM}(x_{t_0-1}) > i$*, and*
(3) $\mathrm{OM}(y_{t_0-1}) < i - \varepsilon n$*.*

Then with at least a constant (that is, $\Omega(1)$) probability we get $x_t > i + \varepsilon n$ before we generate an offspring with i one-bits.

Without proof we note that such iteration t_0 exists with probability $1 - e^{-\Omega(n)}$, which is shown in the supplementary material [1].

Proof. Assume that, at some iteration t', we have $\mathrm{OM}(x_{t-1}) = i + k$ for some $k \in [1..\varepsilon n]$. For all $t \ge t'$ let A_t be an event that we either have $\mathrm{OM}(x_t) > i + k$ or we generate an offspring with exactly i one-bits in generation t. Let B_t be an event that we have $\mathrm{OM}(x_t) > i + k$. Let also C_t be $\bigcap_{j=t'}^{t-1} \overline{A_j}$. Then by Lemma 7 and since B_t is a sub-event of A_t, the probability p_k that we get $\mathrm{OM}(x_t) > i + k$ before we cover the fitness value $(i, n-i)$ is at least

$$\begin{aligned}\Pr[B_t \mid A_t \cap C_t] &= \frac{\Pr[B_t \mid C_t]}{\Pr[A_t \mid C_t]} = \frac{\Pr[B_t \mid C_t]}{\Pr[B_t \cup (A_t \setminus B_t) \mid C_t]} \\ &= \frac{\Pr[B_t \mid C_t]}{\Pr[B_t \mid C_t] + \Pr[A_t \setminus B_t \mid C_t]} = \frac{1}{1 + \frac{\Pr[A_t \setminus B_t \mid C_t]}{\Pr[B_t \mid C_t]}}.\end{aligned}$$

Event $A_t \setminus B_t$ conditional on C_t is the event when we create an individual with exactly i one-bits. If we chose y_{t-1} as a parent, then to do this we would need to flip at least εn bits, the probability of which is $e^{-\Omega(n)}$ by Chernoff bounds. If we choose x_{t-1} as a parent, then we need to flip at least k one-bits, the probability of which is at most $\binom{i+k}{k}(\frac{1}{n})^k$ by Lemma 1.10.37 in [6]. Consequently, we have

$$\begin{aligned}\Pr[A_t \setminus B_t \mid C_t] &\le \frac{1}{2} \cdot e^{-\Omega(n)} + \frac{1}{2} \cdot \binom{i+k}{k} \left(\frac{1}{n}\right)^k \\ &\le \frac{e^{-\Omega(n)} + \frac{n^k}{k! n^k}}{2} = \frac{e^{-\Omega(n)} + \frac{1}{k!}}{2}.\end{aligned}$$

The probability of B_t conditional on C_t is at least the probability that we chose x_{t-1} as a parent and flip only one zero-bit in it, that is,

$$\Pr[B_t \mid C_t] \ge \frac{1}{2} \cdot \frac{n-i-k}{n} \left(1 - \frac{1}{n}\right)^{n-1} \ge \frac{n - \frac{n}{2} - 5\varepsilon n}{2en} = \frac{1-10\varepsilon}{2e}.$$

Hence, we have

$$\Pr[B_t \mid A_t \cap C_t] \ge \frac{1}{1 + \frac{e^{-\Omega(n)} + \frac{1}{k!}}{2} \cdot \frac{2e}{1-10\varepsilon}} = \frac{1}{1 + c\left(e^{-\Omega(n)} + \frac{1}{k!}\right)},$$

where $c = \frac{e}{1-10\varepsilon} = \Omega(1)$, if $\varepsilon < \frac{1}{10}$.

The probability that we reach $\mathrm{OM}(x_t) > i + \varepsilon n$ before we cover the hole is at least the probability that for each OneMax value visited by x_t we increase this value before we cover the hole. By the law of total probability used inductively over all values of k from 1 to εn, this probability is at least

$$\begin{aligned}\prod_{k=1}^{\varepsilon n} \frac{1}{1 + c\left(e^{-\Omega(n)} + \frac{1}{k!}\right)} &= \frac{1}{\exp\left(\ln \prod_{k=1}^{\varepsilon n}\left(1 + c\left(e^{-\Omega(n)} + \frac{1}{k!}\right)\right)\right)} \\ &= \frac{1}{\exp\left(\sum_{k=1}^{\varepsilon n} \ln\left(1 + c\left(e^{-\Omega(n)} + \frac{1}{k!}\right)\right)\right)} \\ &\ge \frac{1}{\exp\left(\sum_{k=1}^{\varepsilon n} c\left(e^{-\Omega(n)} + \frac{1}{k!}\right)\right)} \\ &\ge \frac{1}{\exp\left(c\varepsilon n e^{-\Omega(n)} + ce\right)} = \frac{1}{e^{ce + o(1)}} = \Omega(1).\end{aligned}$$

□

We are now in position to prove the main result of this section, Theorem 6.

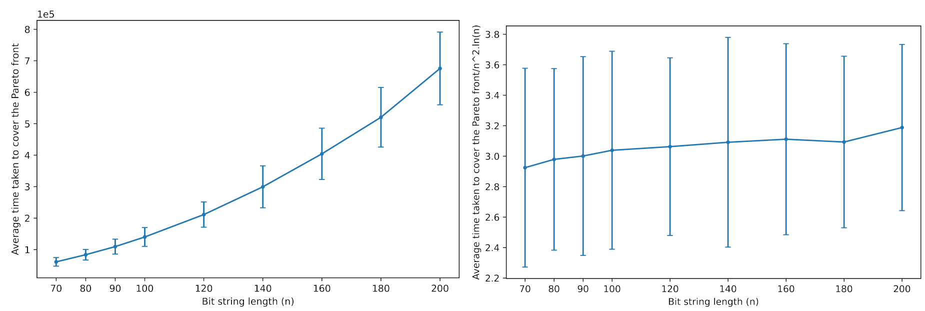
Proof (Proof of Theorem 6). By Lemma 8, assuming that with high probability its conditions are satisfied at some iteration t_0, with probability at least $\Omega(1)$, a run of GSEMO$_2$ is in a situation where there is some fitness value $(i, n-i)$ which is not present in the population, $\mathrm{OM}(x_t) > i + \varepsilon n$ and $\mathrm{OM}(y_t) < i - \varepsilon n$. Therefore, in all consequent iterations, to generate an individual with exactly i one-bits we need to flip at least εn bits in the parent (independently on which edge point we chose), the probability of which by the Chernoff bound is $e^{-\Omega(n)}$. Hence, the expected time until we cover the whole Pareto front is at least $e^{\Omega(n)}$. Since this happens with at least a constant probability, the total expected run time of the GSEMO$_2$ is also $e^{\Omega(n)}$, that is, it is super-polynomial.

8 Anti-Aligned LFC

We use experimental results to extend our analyses to anti-aligned fitness functions from LFC. For the OMC, many individuals on the Pareto front have many neighbors on the Pareto front, so the exploration of the Pareto front is efficient. For anti-aligned fitness functions from LFC, we are only guaranteed a single neighbor (in each possible direction).

We start our analyses with two anti-aligned LFC functions without any shared bits. We assume, without loss of generality, that the optimum (minimum) of f_w is 1^n and that the weights are sorted based on the absolute value in descending order and thus the optimum of f_v is 0^n and the weights are sorted in ascending order. The lemma below is about how an individual on the Pareto-front looks like while optimizing anti-aligned LFC functions without any shared bits and the proof can be found in the supplementary material [1].

Proposition 9 *Let $w, v \in \mathbb{R}^n$ be such that w has only negative values, v has only positive values and the values are in ascending order. Then the Pareto dimension of the multi-objective function (f_w, f_v) is n and the Pareto front is $\{1^i 0^{n-i} \mid 0 \leq i \leq n\}$.*



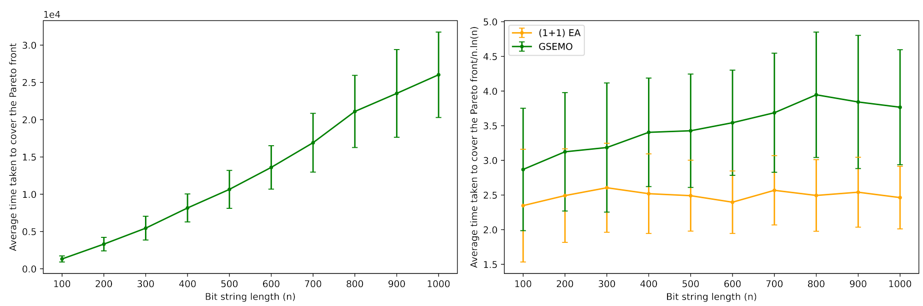
(a) Average run time

(b) Average run time normalized by $n^2 \cdot \ln(n)$

Fig. 1: Average run time of GSEMO on anti-aligned LFC functions with no shared bits.

We now empirically analyze the performance of GSEMO on minimizing two different types of anti-aligned LFC functions. First, we look at (f_w, f_v), where f_w, f_v are two anti-aligned LFC functions without any shared bits. That is, the optimum of f_w and the optimum of f_v differ in each bit position. Let n be the length of the bit string. The values of the weight vectors w and v are randomly chosen from $(-1, 0)$ and $(0, 1)$, respectively and are sorted in ascending order. We consider the mean of 100 independent runs of GSEMO on (f_w, f_v). In Figure 1(a), for each n, we have the mean and the standard deviation of total number of iterations required by the each run of GSEMO to cover the whole Pareto front and in Figure 1(b) this value is divided by $n^2 \cdot \ln(n)$. We can observe from the Figure 1(b), since the mean of the run time is almost a constant line, that GSEMO minimizing two anti-aligned LFC functions without any shared bits appears to have an expected run time of $O(n^2 \cdot \log(n))$.

We next look at (f_w, f_v), where f_w, f_v are two anti-aligned LFC functions without any conflicted bits. That is, f_w and f_v have the same global optimum. The values of the weight vectors w and v are randomly chosen from $(0, 1)$. Without loss of generality, let the weights be positive and the weight vector w be sorted in descending order and the weight vector v be sorted in ascending order. Note that the Pareto front is $\{0^n\}$. In the case of SEMO, the optimization process is similar to the random local search algorithm optimizing the ZeroMax function. Since in each iteration only one bit is flipped, the offspring gets rejected if a 0 bit is flipped to 1 and the offspring replaces the parent if a 1 bit is flipped to 0. This guarantees that SEMO has exactly one individual in the population at each iteration. However, in the case of GSEMO, the population size could be more than one, and the individuals in the population need not have the same number of 0s. We are interested in whether these possibilities slow down search compared with the run time required for the single objective optimization of any of these fitness functions.



(a) Average run time

(b) Average run time of GSEMO and (1+1) EA on f_w normalized by $n \cdot \ln(n)$

Fig. 2: Average run time of GSEMO on anti-aligned LFC functions with no conflicted bits.

We empirically analyze the performance of GSEMO minimizing (f_w, f_v) by considering the mean of 100 independent runs. In Figure 2(a), for each n, we have the mean and standard deviation of total number of iterations required by each run of GSEMO to cover the whole Pareto front and in Figure 2(b) this value is divided by $n \cdot \ln(n)$. In Figure 2(b), we also have the average run time of the (1+1) EA on minimizing f_w, one of the objectives of the two objectives considered for GSEMO. We can observe from the Figure 2(b), that GSEMO on two anti-aligned LFC functions without any conflicted bits appears to have an expected run time of $O(n \cdot \log(n))$, while being a constant factor slower than just on one of the two functions.

References

1. Antipov, D., Kötzing, T., Radhakrishnan, A.: Supplementary material - greedy versus curious parent selection for multi-objective evolutionary algorithms (Apr 2024), https://zenodo.org/records/10990807
2. Antipov, D., Neumann, A., Neumann, F.: Rigorous runtime analysis of diversity optimization with GSEMO on OneMinMax. In: Foundations of Genetic Algorithms, FOGA 2023. p. 3–14. ACM (2023)
3. Brockhoff, D., Friedrich, T., Hebbinghaus, N., Klein, C., Neumann, F., Zitzler, E.: Do additional objectives make a problem harder? In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation GECCO 2007. p. 765–772. ACM (2007)
4. Covantes Osuna, E., Gao, W., Neumann, F., Sudholt, D.: Speeding up evolutionary multi-objective optimisation through diversity-based parent selection. In: Proceedings of the Genetic and Evolutionary Computation Conference GECCO 2017. p. 553–560. ACM (2017)
5. Covantes Osuna, E., Gao, W., Neumann, F., Sudholt, D.: Design and analysis of diversity-based parent selection schemes for speeding up evolutionary multi-objective optimisation. Theor. Comput. Sci. **832**, 123–142 (2018)
6. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. In: Doerr, B., Neumann, F. (eds.) Theory of Evolutionary Computation - Recent Developments in Discrete Optimization, pp. 1–87. Natural Computing Series, Springer (2020)
7. Doerr, B., Doerr, C., Ebel, F.: From black-box complexity to designing new genetic algorithms. Theoretical Computer Science **567**, 87–104 (2015)
8. Doerr, B., Gao, W., Neumann, F.: Runtime analysis of evolutionary diversity maximization for oneminmax. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016 GECCO 2016. p. 557–564. ACM (2016)
9. Doerr, B., Kötzing, T.: Lower bounds from fitness levels made easy. Algorithmica **86**(2), 367–395 (2024)
10. Doerr, B., Qu, Z.: A first runtime analysis of the NSGA-II on a multimodal problem. IEEE Trans. Evol. Comput. **27**(5), 1288–1297 (2023)
11. Doerr, B., Qu, Z.: Runtime analysis for the NSGA-II: provable speed-ups from crossover. In: Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023. pp. 12399–12407. AAAI Press (2023)
12. Doerr, B., Zheng, W.: Theoretical analyses of multi-objective evolutionary algorithms on multi-modal objectives: (hot-off-the-press track at gecco 2021). In: Proceedings of the 2021 Annual Conference on Genetic and Evolutionary Computation GECCO 2021. p. 25–26. ACM (2021)
13. Doerr, C., Lengler, J.: Onemax in black-box models with several restrictions. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation GECCO 2015. p. 1431–1438. ACM (2015)
14. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theor. Comput. Sci. **276**(1-2), 51–81 (2002)
15. Droste, S., Jansen, T., Wegener, I.: Upper and lower bounds for randomized search heuristics in black-box optimization. Theory Comput. Syst. **39**(4), 525–544 (2006)
16. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing, Second Edition. Natural Computing Series, Springer (2015)
17. Friedrich, T., Hebbinghaus, N., Neumann, F.: Plateaus can be harder in multi-objective optimization. In: 2007 IEEE Congress on Evolutionary Computation CEC. pp. 2622–2629 (2007)

18. Friedrich, T., Horoba, C., Neumann, F.: Illustration of fairness in evolutionary multi-objective optimization. Theor. Comput. Sci. **412**(17), 1546–1556 (2011)
19. Giel, O.: Expected runtimes of a simple multi-objective evolutionary algorithm. In: The 2003 Congress on Evolutionary Computation CEC. vol. 3, pp. 1918–1925 Vol.3 (2003)
20. Giel, O., Lehre, P.K.: On the effect of populations in evolutionary multi-objective optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation GECCO 2006. p. 651–658. ACM (2006)
21. Kötzing, T., Krejca, M.S.: First-hitting times under drift. Theoretical Computer Science **796**, 51–69 (2019)
22. Laumanns, M., Thiele, L., Zitzler, E.: Running time analysis of multiobjective evolutionary algorithms on pseudo-Boolean functions. IEEE Transactions on Evolutionary Computation **8**(2), 170–182 (2004)
23. Laumanns, M., Thiele, L., Zitzler, E., Welzl, E., Deb, K.: Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. In: Proceedings of the 7th International Conference on Parallel Problem Solving from Nature PPSN 2002. p. 44–53. Springer-Verlag (2002)
24. Li, Y.L., Zhou, Y.R., Zhan, Z.H., Zhang, J.: A primary theoretical study on decomposition-based multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation **20**(4), 563–576 (2016)
25. Qian, C., Tang, K., Zhou, Z.H.: Selection hyper-heuristics can provably be helpful in evolutionary multi-objective optimization. In: Parallel Problem Solving from Nature PPSN 2016. Springer (2016)
26. Qian, C., Yu, Y., Zhou, Z.H.: An analysis on recombination in multi-objective evolutionary optimization. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation GECCO 2011. p. 2051–2058. ACM (2011)
27. Witt, C.: Tight bounds on the optimization time of a randomized search heuristic on linear functions. Combinatorics, Probability and Computing **22**(2), 294–318 (2013)
28. Zheng, W., Doerr, B.: Mathematical runtime analysis for the non-dominated sorting genetic algorithm II (NSGA-II). Artif. Intell. **325**, 104016 (2023)

A Omitted proofs from Section 4

Theorem 10 *We have* OMC $\subseteq$ LFC*, and both* OMC *and* LFC *are closed under isomorphisms of the hypercube.*

Proof (Proof of Theorem 2 in the paper). For a class C to be closed under isomorphisms of the hypercube means: (a) for $f \in C$ and $\pi : \{0,1\}^n \to \{0,1\}^n$ a permutation of the bit positions, we have $f \circ \pi \in C$; and (b) for $z \in \{0,1\}^n$ and $\oplus$ the bit-wise XOR, we have that $x \mapsto f(x \oplus z)$ is in C. For both classes mentioned it is easy to check that these properties hold.

Regarding OMC $\subseteq$ LFC: Given $a \in \{0,1\}^n$, and let $A = \{i \in [n] \mid a_i = 0\}$ and $B = \{i \in [n] \mid a_i = 1\}$ we let $w_{n+1} = |a|_1 = |B|$ and, for all $i \leq n$, $w_i = 1$ if $i \in A$ and $w_i = -1$ otherwise. Using the indicator function $\mathbf{1}$ we have, for all $x \in \{0,1\}^n$,

$$\begin{aligned}
\mathrm{OM}_a(x) = H(a,x) &= \sum_{i\in[n]} \mathbf{1}[x_i \neq a_i] \\
&= \sum_{i\in A} \mathbf{1}[x_i = 1] + \sum_{i\in B} \mathbf{1}[x_i = 0] \\
&= \sum_{i\in A} 1 \cdot x_i + \sum_{i\in B} (1 - x_i) \\
&= |B| + \sum_{i\in A} 1 \cdot x_i + \sum_{i\in B} (-1) \cdot x_i \\
&= w_{n+1} + \sum_{i\in A} w_i \cdot x_i + \sum_{i\in B} w_i \cdot x_i \\
&= f_w(x).
\end{aligned}$$

B Omitted proofs from Section 6

Corollary 11 *Let $f, g \colon \mathbb{R} \to \mathbb{R}$ be strictly monotone increasing. Then the novelty ranking paired without reset which always chooses an individual that has been considered least number of times (top individual) leads to a run time of $O(n^2)$ on minimizing $(f(OM), g(ZM))$.*

Proof (Proof of Corollary 5 in the paper).

We argue that the optimization process of the algorithm on the function $(f(OM), g(ZM))$ is no different than the algorithm optimizing OneMinMax function. At any time t, let P^t be the population. Then an offspring y created by mutating a parent individual will always get accepted in both OneMinMax and $(f(OM), g(ZM))$ case. The reason is the following, suppose $\forall x \in P^t, |y|_1 \neq |x|_1$. Then y be will get accepted because $(|y|_1, n - |y|_1)$ and $(f(|y|_1), (n - |y|_1))$ will not be dominated by any other individual in the population, while optimizing OneMinMax and $(f(OM), g(ZM))$, respectively. Suppose $\forall x \in P^t, |y|_1 \neq |x|_1$

is not true and $\exists x \in P^t$, such that $|y|_1 = |x|_1$. Then y replaces one of the individuals in the population which has the same fitness (since the algorithm always prefers the new individual) in both OneMinMax and $(f(OM), g(ZM))$ case.

We showed that the algorithm takes $O(n^2)$ to find all the individuals on the Pareto front while optimizing OneMinMax and because of the above mentioned reasons, the same proof follows for $(f(OM), g(ZM))$. □

C Omitted proofs from Section 8

Proposition 12 *Let $w, v \in \mathbb{R}^n$ be such that w has only negative values, v has only positive values and the values are in ascending order. Then the Pareto dimension of the multi-objective function (f_w, f_v) is n and the Pareto front is $\{1^i 0^{n-i} \mid 0 \le i \le n\}$.*

Proof (Proof of Proposition 9 in the paper). Let $S = \{1^i 0^{n-i} \mid 0 \le i \le n\}$. To show that the set S is the Pareto front, we need to show that S has only incomparable individuals and every individual in $\{0,1\}^n \setminus S$ is dominated by an individual in S.

First we show that S has only incomparable individuals. Let x, y be any two distinct individuals in S and without loss of generality, let $|x|_1 > |y|_1$. Since w has only negative values, more 1s in x lead to less f_w function value, therefore $f_w(x) < f_w(y)$. Similarly, since v has only positive values, more 1s in x lead to more f_v function value, which in turn implies $f_v(x) > f_v(y)$. Thus x and y are not comparable and S has only incomparable individuals.

Let $z \in \{0,1\}^n \setminus S$ and $z' = 1^{|z|_1} 0^{|z|_0}$. We claim that, z is dominated by the individual $z' \in S$. Since the weights of f_w are negative and ordered based on the absolute value in descending order more 1s in the beginning bit positions of the individual are beneficial. Similarly, since f_v has only positive weights and the weights are in ascending order, more 0s in the tail bit positions of the individual are beneficial. Note that z and z' have same number of 1s but z has non-leading 1s, which implies $f_w(z) > f_w(z')$ and z also has non-trailing 0s, which implies $f_v(z) > f_v(z')$. Therefore z' dominates z. □

D Omitted proofs from Section 7

In the results shown in this section we use the following auxiliary lemma.

Lemma 13 (Improvement probability) *Let z be a bit string with exactly i one-bits and let X be a random variable following a binomial distribution $\mathrm{Bin}(n-i, \frac{1}{n})$. Then, for any $k \in \mathbb{N}$, the probability that the standard bit mutation applied to z creates a bit string with at least $i+k$ one-bits is at least $\frac{1}{e}\Pr[X = k]$ and at most $\frac{(k+1)(1-\frac{1}{n})}{k+\frac{i-1}{n}}\Pr[X = k]$.*

Proof. To show the lower bound we simply argue that flipping exactly k zero-bits and no one-bits gives us a bit string with exactly $i+k$ one-bits, and therefore, it is a sub-event of getting a bit string with at least $i+k$ one-bits. The probability of it is

$$\binom{n-i}{k}\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{n-i-k}\cdot\left(1-\frac{1}{n}\right)^i \geq \Pr[X=k]\cdot\frac{1}{e}.$$

To show the upper bound, we follow exactly the same arguments as in Lemma 12 in [9] and show that for all $\ell \geq k$ the probability to get a bit string with exactly $i+\ell$ one-bits is at most $\Pr[X=\ell]$. By the union bound over all $\ell \geq k$ we conclude that the probability to get a bit string with at least $i+k$ one-bits is at most $\Pr[X \geq k]$. Finally, by Lemma 1.10.38 in [6] we get that this is at most $\frac{(k+1)(1-\frac{1}{n})}{k+\frac{i-1}{n}}\Pr[X=k]$. □

The following Lemmas 14 and 16 show that with probability at least $1-e^{-\Omega(n)}$ conditions of Lemma 8 are satisfied at some iteration of the GSEMO$_2$.

Lemma 14 *Consider a run of the GSEMO$_2$ on* OneMinMax. *For an arbitrary small constant $\varepsilon > 0$, with probability $1-e^{-\Omega(n)}$, at some iteration τ we get a population in which*

(a) $\mathrm{OM}(x_\tau) - \mathrm{OM}(y_\tau) \geq \varepsilon n$ *and*
(b) $\mathrm{OM}(x_\tau) \leq \frac{n}{2} + 3\varepsilon n$.

Proof. Let x_0 be the initial search point, which is generated uniformly at random. By the definition of the GSEMO$_2$, the fitness of x_0 will always stay in the population, hence we will always have $\mathrm{OM}(y_t) \leq \mathrm{OM}(x_0) \leq \mathrm{OM}(x_t)$. By Chernoff bounds (see, e.g., Theorem 1.10.1 in [6]), we have that $\frac{n}{2}-\varepsilon n \leq \mathrm{OM}(x_0) \leq \frac{n}{2}+\varepsilon n$ with probability $1-e^{-\Omega(n)}$. We now condition on this event.

Let τ be the first iteration where we have $\mathrm{OM}(x_\tau) \geq \frac{n}{2}+2\varepsilon n$ (note that the probability that there is no such iteration is zero). Then we have

$$\mathrm{OM}(x_\tau) - \mathrm{OM}(y_\tau) \geq \mathrm{OM}(x_\tau) - \mathrm{OM}(x_0) \geq \left(\frac{n}{2}+2\varepsilon n\right) - \left(\frac{n}{2}+\varepsilon n\right) = \varepsilon n,$$

hence at iteration τ condition (a) is satisfied.

To estimate the probability that $\mathrm{OM}(x_\tau) \leq \frac{n}{2}+3\varepsilon n$, we use Lemma 7, where A_t is the event that in iteration t we generate an offspring z_t with $\mathrm{OM}(z_t) \geq \frac{n}{2}+2\varepsilon n$, and B_t is the event that z_t is such that $\mathrm{OM}(z_t) \geq \frac{n}{2}+3\varepsilon n$. Hence, to use Lemma 7 we need to find an upper bound over all t on the probability of B_t conditional on A_t and all $\overline{A_j}$ with $j \in [1..t-1]$, that is, on $A_t \cap C_t$.

Consider some arbitrary iteration t and let m be the number of one-bits in the individual chosen as a parent in this iteration. Condition $\cap_{j=1}^{t-1}\overline{A_j}$ implies that $m < \frac{n}{2}+2\varepsilon n$. By the law of total probability we have

$$\Pr[B_t \mid A_t \cap C_t] = \sum_{i=0}^{\lceil \frac{n}{2}+2\varepsilon n\rceil - 1} \Pr[m=i]\Pr[B_t \mid A_t \cap C_t \cap (m=i)]$$

$$
\begin{aligned}
&\leq \max_{\ell < \frac{n}{2} + 2\varepsilon n} \Pr[B_t \mid A_t \cap C_t \cap (m = i)] \\
&= \max_{\ell < \frac{n}{2} + 2\varepsilon n} \Pr[B_t \mid A_t \cap (m = i)],
\end{aligned}
$$

since $(m = i)$ is a sub-event of C_t for all $i < \frac{n}{2} + 2\varepsilon n$. Since B_t is a sub-event of A_t, we have

$$
\Pr[B_t \mid A_t \cap (m = i)] = \frac{\Pr[B_t \cap A_t \mid m = i]}{\Pr[A_t \mid m = i]} = \frac{\Pr\left[\mathrm{OM}(z_t) \geq \frac{n}{2} + 3\varepsilon n \mid m = i\right]}{\Pr\left[\mathrm{OM}(z_t) \geq \frac{n}{2} + 2\varepsilon n \mid m = i\right]}.
$$

By Lemma 13 we have

$$
\begin{aligned}
\Pr\left[\mathrm{OM}(z_t) \geq \frac{n}{2} + 3\varepsilon n \mid m = i\right] \leq \;& \frac{\left(\frac{n}{2} + 3\varepsilon n - i + 1\right)\left(1 - \frac{1}{n}\right)}{\frac{n}{2} + 3\varepsilon n - i + \frac{i}{n}} \\
& \cdot \Pr\left[\mathrm{Bin}\left(n - i, \frac{1}{n}\right) = \frac{n}{2} + 3\varepsilon n - i\right].
\end{aligned}
$$

Denoting $(\frac{n}{2} + 3\varepsilon n - i)$ by k (similar to the notation in Lemma 13), we note that the fraction in front of the probability is descending in k. Since $k \geq \varepsilon n = \Omega(n)$ by the conditions on i, we have that this fraction is at most

$$
\frac{(k+1)\left(1 - \frac{1}{n}\right)}{k + \frac{i}{n}} \leq \frac{k+1}{k} = 1 + o(1),
$$

hence we have

$$
\Pr\left[\mathrm{OM}(z_t) \geq \frac{n}{2} + 3\varepsilon n \mid m = i\right] \leq (1 + o(1)) \Pr\left[\mathrm{Bin}\left(n - i, \frac{1}{n}\right) = \frac{n}{2} + 3\varepsilon n - i\right].
$$

By Lemma 13 we also have

$$
\Pr\left[\mathrm{OM}(z_t) \geq \frac{n}{2} + 2\varepsilon n \mid m = i\right] \geq \frac{1}{e} \Pr\left[\mathrm{Bin}\left(n - i, \frac{1}{n}\right) = \frac{n}{2} + 2\varepsilon n - i\right].
$$

Therefore, using the estimates for binomial coefficients from Lemma 1.4.9 in [6], we compute the upper bound on $\Pr[B_t \mid A_t \cap (m = i)]$ as follows.

$$
\begin{aligned}
\Pr[B_t \mid A_t \cap (m = i)] &\leq (1 + o(1))e \cdot \frac{\Pr\left[\mathrm{Bin}\left(n - i, \frac{1}{n}\right) = \frac{n}{2} + 3\varepsilon n - i\right]}{\Pr\left[\mathrm{Bin}\left(n - i, \frac{1}{n}\right) = \frac{n}{2} + 2\varepsilon n - i\right]} \\
&= (1 + o(1))e \cdot \frac{\binom{n-i}{\frac{n}{2} + 3\varepsilon n - i}\left(\frac{1}{n}\right)^{\frac{n}{2} + 3\varepsilon n - i}\left(1 - \frac{1}{n}\right)^{\frac{n}{2} - 3\varepsilon n}}{\binom{n-i}{\frac{n}{2} + 2\varepsilon n - i}\left(\frac{1}{n}\right)^{\frac{n}{2} + 2\varepsilon n - i}\left(1 - \frac{1}{n}\right)^{\frac{n}{2} - 2\varepsilon n}} \\
&\leq (1 + o(1))e^2 \cdot \left(\frac{1}{n}\right)^{\varepsilon n} \cdot \frac{\left(\frac{e(n-i)}{\frac{n}{2} + 3\varepsilon n - i}\right)^{\frac{n}{2} + 3\varepsilon n - i}}{\left(\frac{n-i}{\frac{n}{2} + 2\varepsilon n - i}\right)^{\frac{n}{2} + 2\varepsilon n - i}}
\end{aligned}
$$

$$= (1+o(1))e^2 \cdot \left(\frac{n-i}{n}\right)^{\varepsilon n} \cdot \left(\frac{e\left(\frac{n}{2}+2\varepsilon n-i\right)}{\left(\frac{n}{2}+3\varepsilon n-i\right)}\right)^{\left(\frac{n}{2}+3\varepsilon n-i\right)} \cdot \left(\frac{1}{\left(\frac{n}{2}+2\varepsilon n-i\right)}\right)^{\varepsilon n}.$$

Note that $(\frac{n-i}{n})^{\varepsilon n} \le (\frac{1}{2}+2\varepsilon)^{\varepsilon n} = e^{-\Omega(n)}$. To estimate the remaining two terms we again denote $k = (\frac{n}{2}+3\varepsilon n - i) > \varepsilon n$, so that we need to find an upper bound on

$$\left(\frac{e(k-\varepsilon n)}{k}\right)^k \cdot \left(\frac{1}{k-\varepsilon n}\right)^{\varepsilon n}.$$

Case 1: if $k \ge e(k-\varepsilon n)$, then both fractions inside the brackets are at most one and their exponents are positive, therefore, their product is also at most one.

Case 2: if $k < e(k-\varepsilon n)$, then we have $k > \frac{e\varepsilon n}{e-1}$. Since we also have $k \le n$ by the definition of k, we have

$$\left(\frac{e(k-\varepsilon n)}{k}\right)^k \cdot \left(\frac{1}{k-\varepsilon n}\right)^{\varepsilon n} \le e^n \cdot \left(\frac{e-1}{\varepsilon n}\right)^{\varepsilon n} = \left(\frac{e(e-1)^\varepsilon}{\varepsilon^\varepsilon n^\varepsilon}\right)^n.$$

If n is large enough (namely, $n \ge \frac{e^{1/\varepsilon}(e-1)}{\varepsilon}$), then this expression is at most one.

Bringing these two cases together, we obtain that the probability $\Pr[B_t \mid A_t]$ is at most $e^{-\Omega(n)}$. Recalling that we start with $\mathrm{OM}(x_0) \le \frac{n}{2}+\varepsilon n$ with probability $1-e^{-\Omega(n)}$, both lemma conditions (a) and (b) are satisfied at iteration τ with probability $1-e^{-\Omega(n)}$. $\square$

We now show that, under the conditions of Lemma 14, we are likely to make a *hole* in our population, that is, to jump over one fitness value, before we increase x_t by εn. We first show a situation when such a hole occurs with constant probability and then show that this situation is likely to happen during a typical run of the algorithm.

Lemma 15 *Consider a run of the GSEMO*$_2$ *on* OneMinMax. *Consider some iteration* t_0*, which starts with* $\mathrm{OM}(x_{t_0-1}) = i$*, where* i *is in* $[\frac{n}{2}+2\varepsilon n, \frac{n}{2}+4\varepsilon n]$*, and* $\mathrm{OM}(y_{t_0-1}) \le \frac{n}{2}+\varepsilon n$*. Let* $\tau \ge t_0$ *be the first iteration when we get* $\mathrm{OM}(x_\tau) > i$*. Then with probability at least* $\frac{(1-8\varepsilon)^2}{16e} - o(1)$ *we have* $\mathrm{OM}(x_\tau) > i+1$*.*

Proof. We use Lemma 7 to prove this result. For this we define A_t as an event when $\mathrm{OM}(x_t) > i$, and B_t as an event when $\mathrm{OM}(x_t) > i+1$. Then τ in the condition of this lemma is exactly the same as in Lemma 7. Thus, to find the probability that $\mathrm{OM}(x_\tau) > i+1$, we aim at bounding the conditional probability $\Pr[B_t \mid A_t \cap C_t]$, where $C_t = \bigcap_{j=t_0}^{t-1} \overline{A_t}$. Similar to the proof of Lemma 14, since B_t is a sub-event of A_t, we compute

$$\Pr[B_t \mid A_t \cap C_t] = \frac{\Pr[B_t \mid C_t]}{\Pr[A_t \mid C_t]} \ge \Pr[B_t \mid C_t].$$

Event C_t implies that in the start of iteration t we have $x_{t-1} = i$. Hence, the probability of B_t is at least the probability that we choose x_{t-1} as a parent and flip exactly 2 zero-bits in it, which is,

$$\frac{1}{2} \cdot \binom{n-i}{2} \left(\frac{1}{n}\right)^2 \left(1 - \frac{1}{n}\right)^{n-2} \geq \frac{1}{4e} \cdot \left(\frac{\frac{n}{2} - 4\varepsilon n - 1}{n}\right)^2 = \frac{(1-8\varepsilon)^2}{16e} - o(1).$$

□

We now show that we make make a hole before reaching $x_t = \frac{n}{2} + 4\varepsilon n$.

Lemma 16 *Consider a run of the GSEMO$_2$ on* OneMinMax*. Assume that we start at iteration τ in which the conditions of Lemma 14 are satisfied. Then, with probability $1-e^{-\Omega(n)}$, there exists an iteration τ' such that, for some $i \leq \frac{n}{2}+4\varepsilon n$, we have $\mathrm{OM}(x_{\tau'-1}) < i$ and $\mathrm{OM}(x_{\tau'}) > i$. That is, the algorithm jumps over the fitness value $(i, n-i)$ in iteration τ'.*

Proof. If there is no such iteration τ', then for each fitness value $j \in [i, \frac{n}{2} + 4\varepsilon n]$, when we got for the first time $\mathrm{OM}(x_t) > j$, it was $\mathrm{OM}(x_t) = j + 1$. For all $j \in [i, \frac{n}{2} + 4\varepsilon n)$ let A_j be the event that at some iteration we have $\mathrm{OM}(x_t) = j$. Then the probability that this event occurs for all j can be inductively written as

$$\Pr\left[\bigcap_{j=i}^{\frac{n}{2}+4\varepsilon n-1} A_j\right] = \Pr[A_i]\Pr\left[\bigcap_{j=i+1}^{\frac{n}{2}+4\varepsilon n-1} A_j \;\middle|\; A_i\right] = \prod_{k=i}^{\frac{n}{2}+4\varepsilon n-1} \Pr\left[A_k \;\middle|\; \bigcap_{j=i}^{k-1} A_j\right].$$

The probability of A_k conditional on all previous A_j is the probability that when we have $x_t = k-1$, the next increment of x_t will be by one. By Lemma 15, this probability for all k is at most $1 - \frac{(1-8\varepsilon)^2}{16e} + o(1)$, therefore, we have

$$\Pr\left[\bigcap_{j=i}^{\frac{n}{2}+4\varepsilon n-1} A_j\right] \leq \left(1 - \frac{(1-8\varepsilon)^2}{16e} + o(1)\right)^{\varepsilon n} = e^{-\Omega(n)},$$

which proves the lemma. □