Proven Approximation Guarantees in Multi-Objective Optimization: SPEA2 Beats NSGA-II

Yasser Alghouass[1], **Benjamin Doerr**[2], **Martin S. Krejca**[2] and **Mohammed Lagmah**[1]
[1]École Polytechnique, Institut Polytechnique de Paris
[2]Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institut Polytechnique de Paris
{firstname.lastname}@polytechnique.edu

Abstract

Together with the NSGA-II and SMS-EMOA, the *strength Pareto evolutionary algorithm 2* (SPEA2) is one of the most prominent dominance-based multi-objective evolutionary algorithms (MOEAs). Different from the NSGA-II, it does not employ the crowding distance (essentially the distance to neighboring solutions) to compare pairwise non-dominating solutions but a complex system of σ-distances that builds on the distances to all other solutions. In this work, we give a first mathematical proof showing that this more complex system of distances can be superior. More specifically, we prove that a simple steady-state SPEA2 can compute optimal approximations of the Pareto front of the OneMinMax benchmark in polynomial time. The best proven guarantee for a comparable variant of the NSGA-II only assures approximation ratios of roughly a factor of two, and both mathematical analyses and experiments indicate that optimal approximations are not found efficiently.

1 Introduction

Many optimization problems in practice consist of several conflicting objectives. One common approach for such problems is to compute a set of solutions witnessing the Pareto front (or a sufficiently diverse subset thereof) and then let a human decision maker select the final solution. For such multi-objective optimization problems, evolutionary algorithms with their population-based nature are an obvious choice, and in fact, such *multi-objective evolutionary algorithms (MOEAs)* are among the most successful algorithms [Coello *et al.*, 2007; Zhou *et al.*, 2011].

Most research on randomized search heuristics is empirical, and much fewer theoretical works exist [Neumann and Witt, 2010; Auger and Doerr, 2011; Jansen, 2013; Zhou *et al.*, 2019; Doerr and Neumann, 2020]. However, theoretical analyses of MOEAs, usually proving runtime guarantees and from this aiming at a deeper understanding of these algorithms, exist for more than twenty years [Laumanns *et al.*, 2002; Giel, 2003; Thierens, 2003]. This field has made a huge step forward recently with the first mathematical runtime analysis of the NSGA-II [Zheng *et al.*, 2022; Zheng and Doerr, 2023], the most prominent MOEA. This work was quickly followed up by more detailed analyses of the NSGA-II [Bian and Qian, 2022; Doerr and Qu, 2023a; Doerr and Qu, 2023b; Doerr and Qu, 2023c; Dang *et al.*, 2023; Cerf *et al.*, 2023; Dang *et al.*, 2024; Deng *et al.*, 2024; Zheng and Doerr, 2024a; Zheng and Doerr, 2024b; Doerr *et al.*, 2025] and by analyses of other prominent MOEAs… such as the NSGA-III [Wietheger and Doerr, 2023; Opris *et al.*, 2024; Deng *et al.*, 2025; Opris, 2025], SMS-EMOA [Bian *et al.*, 2023; Zheng and Doerr, 2024c; Li *et al.*, 2025], and SPEA2 [Ren *et al.*, 2024].

Interestingly, these results show very similar performance guarantees for these algorithms (which agree with the results known for the classic (G)SEMO analyzed in the early theoretical works on MOEAs). Also, [Ren *et al.*, 2024] provide a mathematical framework allowing to prove comparable runtime bounds for several MOEAs. The sole outlier so far is the result [Zheng and Doerr, 2024b] showing that the NSGA-II has enormous difficulties for three or more objectives (see [Doerr *et al.*, 2024] for a second such result). This problem of the NSGA-II can be resolved with an additional tie-breaker [Doerr *et al.*, 2025] and then the same performance guarantees as known for the other algorithms hold.

In this work, we detect a second notable performance difference, namely in the ability to approximate the Pareto front. This aspect is understood much less than the time to compute the full Pareto front. The only such result for the algorithms mentioned above is [Zheng and Doerr, 2024a]. In that work, it was argued via theoretical arguments that the classic NSGA-II, computing the crowding distance for all individuals and then selecting the next population, can have difficulties to approximate the Pareto front as it ignores that every removal of an individual affects the crowding distance of other solutions. However, for the algorithm variant that removes the individuals sequentially and updates the crowding distances after each removal (as suggested already in the little known work [Kukkonen and Deb, 2006]), a 2-approximation result was shown for the optimization of the bi-objective OneMinMax benchmark. This is the only approximation guarantee so far for one of the classic algorithms named above.

Our contribution. In this work, we study the approximation ability of the *simple Pareto evolutionary algorithm 2* [Zitzler *et al.*, 2001] (SPEA2). Like the NSGA-II, the SPEA2 is a dominance-based algorithm, that is, its first priority is to keep non-dominated solutions. As a tie-breaker for removing solutions, like the NSGA-II, it regards how close incomparable solutions are and first removes those with other solutions nearby, thus aiming for an evenly spread population. We discuss the precise differences of the distance measures used by the NSGA-II and SPEA2 later in this work, and remark for now only that the SPEA2 takes into account the distances to all other solutions (in the order from near to far), whereas the crowding distance of the NSGA-II only accounts for the distances to the two neighboring solutions in each objective.

As we shall show in this work, with its more complex distance measure, the SPEA2 is able to compute much better approximations of the Pareto front. Taking again the OneMinMax benchmark as example, we prove that a simple version of the SPEA2 finds an optimal approximation to the Pareto front in polynomial time, more precisely, expected time $O(\mu^2 \log(\mu) n \log(n))$ (Theorem 1), where μ is the population size of the SPEA2 and n is the problem size of the benchmark. We contrast this result by showing that there are states in the NSGA-II that are close to being an optimal approximation, but the algorithm takes nonetheless with overwhelming probability a super-polynomial time to get to the optimal approximation (Theorem 11). This shows that the NSGA-II can struggle immensely to compute an optimal approximation whereas the SPEA2 does not have this problem.

These results give a strong evidence for the hypothesis that the more complex way of measuring the distance between solutions of the SPEA2 is worth the additional complexity and leads to significantly stronger approximation abilities.

2 Preliminaries

Denote the natural numbers by $\mathbb{N}$ (with 0) and the reals by $\mathbb{R}$. For all $m, n \in \mathbb{N}$, let $[m..n] := [m, n] \cap \mathbb{N}$ and $[n] := [1..n]$.

For all $n \in \mathbb{N}_{\geq 1}$, we call $x \in \{0,1\}^n$ an *individual*. We call a multi-set of individuals a *population*, and we use standard set operations for populations, which extend naturally to multi-sets. We consider pseudo-Boolean maximization of bi-objective functions $f\colon \{0,1\}^n \to \mathbb{R}^2$, which map individuals to *objective values*. For each $x \in \{0,1\}^n$ and $i \in [2]$, we denote the objective value i of x by $f_i(x)$.

We compare objective values $u, v \in \mathbb{R}$ via the *dominance* partial order. We say u *weakly dominates* v (written $u \succeq v$) if and only if $u_1 \geq v_1$ and $u_2 \geq v_2$, and u *strictly* dominates v if and only if at least one of these inequalities is strict. If and only if neither $u \succeq v$ nor $v \succeq u$, we say u and v are *incomparable*. Given a bi-objective function, we extend this notion to all individuals by implicitly referring to their objective value.

For a (multi-)set P of individuals and a bi-objective function f, we say that an individual $x \in P$ is *non-dominated* if and only if there is no $y \in P$ that strictly dominates u. We call each non-dominated individual of $\{0,1\}^n$ *Pareto-optimal* and its objective value a *Pareto optimum*. Last, we call the set of all Pareto optima the *Pareto front (of f)*.

We use traditional set notation for both normal sets (with unique elements) and multi-sets. The union of multi-sets does not remove any duplicates, and the cardinality of a multi-set accounts for all duplicates. However, if we apply a function to multi-set, then the result is a normal function, that is, the function values are not duplicated. Note that populations are multi-sets whereas sets of objective values are normal sets.

The OneMinMax Benchmark. The OneMinMax (OMM) benchmark, introduced by [Giel and Lehre, 2010], is a bi-objective function that aims at maximizing the number of ones and the number of zeros of an individual. Formally, for all $x \in \{0,1\}^n$, we have

$$\mathrm{OMM}(x) = \Big(\textstyle\sum_{i\in[n]} x_i, \sum_{i\in[n]}(1 - x_i)\Big).$$

Since the objectives are the inverse of each other, no individual strictly dominates another one. Thus, the Pareto front is the set of all objective values, namely $\{(i, n-i) \mid i \in [0..n]\}$, which has a size of $n+1$.

We call the objective values $(0, n)$ and $(n, 0)$ the *extreme* objective values, since they feature the maximum and minimum possible value in each of their objectives.

Optimal spread. We aim at approximating the Pareto front on OMM algorithmically via a population. To this end, let P be a population of size $\mu := |P| \in [2..n]$, and let $\big(\mathrm{OMM}_1(x)\big)_{x\in P} =: (v_i)_{i\in[\mu]}$ be the first objective of each individual in P in ascending order. Furthermore, assume that $v_1 = 0$ and that $v_\mu = n$, that is, that the extreme objective values are witnessed by P. We say that P computes an *optimal spread* on the Pareto front of OMM if and only if for all $i \in [\mu - 1]$, we have $v_{i+1} - v_i \in \{\lfloor \frac{n}{\mu-1} \rfloor, \lceil \frac{n}{\mu-1} \rceil\}$. In other words, the distance between two neighboring objective-values is as close as possible to being equidistant.

In order to more concisely describe an optimal spread, let

$$\begin{aligned} &\alpha := \lfloor \tfrac{n}{\mu-1} \rfloor \text{ and } \beta \in [0..\mu-1] \qquad (1)\\ &\text{such that } \alpha\beta + (\alpha+1)(\mu-1-\beta) = n, \end{aligned}$$

where we note that β is uniquely determined.

3 The SPEA2 and NSGA-II Algorithms

The *strength Pareto evolutionary algorithm 2* [Zitzler *et al.*, 2001] (SPEA2, Algorithm 1) and the *non-dominated sorting genetic algorithm II* [Deb *et al.*, 2002] (NSGA-II, Algorithm 2) are both popular population-based multi-objective optimization heuristics. Since we analyze the SPEA2 in more depth in this paper, we explain it in more detail below in Section 3.1, noting that both algorithms act identically in broad parts on the OMM benchmark. We outline the most important parts of the NSGA-II in Section 3.2, referring for more details to the original paper by [Deb *et al.*, 2002].

3.1 The SPEA2

We regard an equivalent formulation of the SPEA2 that was proposed in the extension of [Wietheger and Doerr, 2024] (currently only on arXiv). The SPEA2 (Algorithm 1) maximizes a given bi-objective problem $f\colon \{0,1\}^n \to \mathbb{R}^2$ by iteratively refining a multi-set of individuals (the *parent population*) of given population size $\mu \in \mathbb{N}_{\geq 1}$. In each iteration, the algorithm generates a user-defined amount $\lambda \in \mathbb{N}_{\geq 1}$ of new individuals (the *offspring population*) via a process called *mutation*. Afterward, the SPEA2 selects the μ most promising individuals among the parent and the offspring population, to be used in the next iteration. The basis for this new population are all non-dominated individuals. If this number is greater than μ, then the algorithm removes individuals based on a clustering technique called *σ-distances*. This is the predominant case in our analysis in Section 4, as the OMM benchmark only features non-dominated objective values.

If, instead, the number of non-dominated individuals is less than μ, then the SPEA2 adds dominated individuals, based on their *strength-based indicator*. Last, if the number of non-dominated individuals is exactly μ, the algorithm proceeds immediately with the next iteration.

Mutation. We consider two types of mutation, each of which is given a *parent* $x \in \{0,1\}^n$ and generates a *new* individual $y \in \{0,1\}^n$ (the *offspring*).

1-bit mutation copies x and flips exactly one bit at position $i \in [n]$ chosen uniformly at random. That is, we have $y_i = 1 - x_i$, and for all $j \in [n] \smallsetminus \{i\}$, we have $y_i = x_i$.

Standard bit mutation copies x and flips each position independently with probability $\frac{1}{n}$. That is, for all $i \in [n]$, we have $\Pr[y_i = 1 - x_i] = \frac{1}{n}$ and $\Pr[y_i = x_i] = 1 - \frac{1}{n}$, independent from all other random choices.

The σ-distances. Given a population P of size $s \in \mathbb{N}_{\geq 1}$ of non-dominated individuals, the σ-distances are based on a function $\sigma\colon \{0,1\}^n \to \mathbb{R}^{s-1}$ that assigns each individual $x \in P$ a vector that contains the Euclidean distances of the objective value of x to those of all other individuals in P in ascending order (breaking ties arbitrarily), which we call the σ-distance of x. We follow the convention of [Zitzler *et al.*, 2001] and denote for all $i \in [s-1]$ the distance of x to its i-closest neighbor by σ_x^i.

In algorithm 1, individuals are removed with respect to the lexicographic ascending order of their σ-distance, breaking ties uniformly at random, and removing the individuals at the beginning of this order. In other words, individuals that are too close to other individuals are removed first. We remark that the σ-distances are updated after each removed individual.

We note that for OMM, we always have $s = \mu + \lambda$, as all individuals are Pareto-optimal. Moreover, we note that for the removal based on σ-distances, only the relative order matters. Since OMM has complementary objectives, it is thus sufficient to consider a distance based on a single objective that has the same monotonicty as the original σ-distances. We focus in our analysis on the first objective, that is, the number of 1s in an individual, for the σ-distances.

Strength-based indicator. We note that the case in which the SPEA2 relies on the strength-based indicator never occurs on OMM, as all individuals in OMM are Pareto-optimal. Hence, we do not explain this operation in detail but refer to the original work by [Zitzler *et al.*, 2001] instead. Roughly, the strength-based indicator assigns each individual x in the combined parent and offspring population a natural number that is the sum over all individuals y that strictly dominate x, adding the number of individuals that y weakly dominates.

Steady-state variant. If $\lambda = 1$, we say that the SPEA2 is *steady-state*. When optimizing OMM, this means that a single individual in algorithm 1 is removed, as all individuals are Pareto-optimal. The case in algorithm 1 is never executed.

Algorithm 1: The strength Pareto evolutionary algorithm 2 [Zitzler *et al.*, 2001] (SPEA2) with parent population size $\mu \in \mathbb{N}_{\geq 1}$, and offspring population size $\lambda \in \mathbb{N}_{\geq 1}$, maximizing a given bi-objective function $f\colon \{0,1\}^n \to \mathbb{R}^2$.

```
1  $t \leftarrow 0$;
2  $P_t \leftarrow \lambda$ independent samples from $\{0,1\}^n$ with
     replacement, chosen uniformly at random (u.a.r.);
3  while termination criterion not met do
4      $Q_t \leftarrow \emptyset$;
5      for $i \in [\lambda]$ do
6          $x \leftarrow$ individual in $P_t$ chosen u.a.r.;
7          $y \leftarrow$ mutate $x$ (see Section 3);
8          $Q_t \leftarrow Q_t \cup \{y\}$;
9      $P_{t+1} \leftarrow$ non-dominated individuals in $P_t \cup Q_t$;
10     if $|P_{t+1}| > \mu$ then
11         iteratively remove an individual in $P_{t+1}$ with
             the smallest $\sigma$-distance until $|P_{t+1}| = \mu$;
12     else if $|P_{t+1}| < \mu$ then
13         iteratively add an individual from
             $(P_t \cup Q_t) \smallsetminus P_{t+1}$ to $P_{t+1}$ with the smallest
             strength-based indicator until $|P_{t+1}| = \mu$;
14     $t \leftarrow t + 1$;
```

3.2 The NSGA-II

Like the SPEA2, the NSGA-II (Algorithm 2) maximizes a given bi-objective function $f\colon \{0,1\}^n \to \mathbb{R}^2$ by maintaining a parent population of $N \in \mathbb{N}_{\geq 1}$ individuals, from which it generates N offspring each iteration, using a mutation operator. Afterward, it selects individuals greedily based on the number of individuals by which they are strictly dominated, starting with the non-dominated individuals. During this phase, all individuals with an identical number are selected. The smallest number where this leads to selecting at least N individuals in total is known as the *critical rank*. For OMM, this means that the entire combined parent and offspring population (of size $2N$) is kept, as all individuals are not strictly dominated, thus all land in the critical rank.

Afterward, the algorithm removes individuals sequentially from the critical rank based on their *crowding distance*, until N individuals remain, breaking ties uniformly at random.

Crowding distances. Given a population R, the crowding distance of an individual $x \in R$ is the sum of its crowding distance *per objective*. For each objective $i \in [2]$, the crowding distance of x is based on the (normalized) distance to its two closest neighbors in objective i. That is, let $S_i = (y_i)_{i \in [|R|]}$ denote R in ascending order of objective i. If x is an extreme solution, that is, if $x = y_1$ or $x = y_{|R|}$, then its crowding distance for objective i is plus infinity. Otherwise, if there is an $i \in [2..|R|-1]$ such that $x = y_i$, then the crowding distance of x for objective i is $\big(f(y_{i+1}) - f(y_{i-1})\big) / \big(f(y_{|R|}) - f(y_1)\big)$.

Algorithm 2: The non-dominated sorting genetic algorithm II [Deb *et al.*, 2002] (NSGA-II) with population size $N \in \mathbb{N}_{\geq 1}$, maximizing a given bi-objective function.

```
1  $t \leftarrow 0$;
2  $P_t \leftarrow N$ independent samples of $\{0,1\}^n$ with
     replacement, each chosen u.a.r.;
3  while termination criterion not met do
4      $Q_t \leftarrow$ offspring population of $P_t$ of size $N$;
5      $R_t \leftarrow P_t \cup Q_t$;
6      $(F_j)_{j \in [r]} \leftarrow$ partition of $R_t$ w.r.t. non-dom. ranks;
7      $j^* \leftarrow$ critical rank of $(F_j)_{j \in [r]}$;
8      $P_{t+1} \leftarrow \bigcup_{j \in [j^*]} F_j$;
9      if $|P_{t+1}| > N$ then
10         iteratively remove an individual in $P_{t+1}$
             from $F_{j^*}$ with the smallest crowding distance
             in $F_{j^*}$ until $|P_{t+1}| = N$;
11     $t \leftarrow t + 1$;
```

For OMM, similar to the SPEA2, it is sufficient to only consider the crowding distance in the first objective, as the two objectives are complementary to each other (and the sorting for each objective just results in an inverse order, up to how ties are handled).

We remark that, different from the σ-distances in the SPEA2, the crowding distance is *not* recomputed each time an individual is removed, which is a general flaw in the NSGA-II, as proven by [Zheng and Doerr, 2024a]. In this article, we use a stable sorting algorithm for the crowding distance computation, which is common practice.

Steady-state variant. If the NSGA-II only produces one offspring each iteration (regardless of N), we say that it is *steady-state*. This means that in each iteration, exactly one individual is removed. Since this effectively recomputes the crowding distance after each removal of an individual, this algorithm variant does not have the problems associated with the classic NSGA-II.

4 The SPEA2 Computes an Optimal Spread Efficiently

Our main result is Theorem 1, which shows that the steady-state SPEA2 with 1-bit mutation and $\mu \leq \frac{n}{3}$ computes an optimal spread on OMM after $O(\mu^2 n \log(\mu) \log(n))$ expected function evaluations[1]. This bound is slower by a factor of $\mu \log \mu$ than the time required to find the extreme objective values (Theorem 4), which may be an artifact of our analysis.

Theorem 1. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation and with $\mu \leq \frac{n}{3}$. Then the expected number of objective function evaluations for computing an optimal spread is $O(\mu^2 n \log(\mu) \log(n))$.*

[1]This is asymptotically equal to the number of iterations, as each iteration generates exactly one offspring in the steady-state variant.

In the following, we start with a high-level overview of the proof structure. Afterward, we provide more details, until we prove Theorem 1.

4.1 High-Level Proof Outline for the SPEA2

Due to all individuals being Pareto-optimal for OMM, our analysis only concerns the case that individuals are removed due to algorithm 1 in Algorithm 1. For most of our analysis, we consider the steady-state variant, which implies that exactly one individual is removed. In order to simplify notation, we only consider the first objective when comparing σ-distances, which is feasible for OMM due to the objectives being complementary, as we also mention in the section on σ-distances in Section 3.

No duplicate objective values. First, we show that the SPEA2 quickly reaches a state in which all individuals in the parent population have distinct objective values (Theorem 3), which is possible because the population size μ is strictly smaller than the size $n+1$ of the Pareto front. Moreover, we prove that the parent population quickly includes the extreme objective values and never loses them (Theorem 4). Once in such a state, we change perspectives and no longer consider individuals but instead the empty intervals between the first objective values of all individuals in the parent population. From this perspective, an optimal spread is computed once all intervals are of size $\lfloor \frac{n}{\mu-1} \rfloor$ or $\lceil \frac{n}{\mu-1} \rceil$.

Useful invariants. We continue by proving that the size of all minimum length (empty) intervals (of objective values) never decreases during the run of the algorithm (Lemma 5). In addition, we show that the number of intervals of minimum length also never decreases (Lemma 6), which in itself is already a strong property of the σ-distances of the SPEA-2. We show analogously that the size of all maximum-length intervals (Lemma 6) never increases. These statements are the foundation of our analysis, as they provide well-defined states of the algorithm from which it can only improve.

Afterward, we show that once the size of all minimum length intervals is at least 2, algorithm 1 reduces to the case that either the offspring or its parent is removed (Lemma 7). This drastically decreases the complexity of the cases to consider in the remaining analysis. In addition, we show in Lemma 8 that intervals of minimum length remain at the borders of the interval $[0..n]$ (until the minimum is increased). This helps us locate such intervals later in the analysis.

Since the SPEA2 features a lot of useful properties once the minimum length interval has size at least 2, we prove in Lemma 9 that such a state is reached quickly. From thereon, we take a more detailed look about how the minimum length interval is increased.

Reducing the number of intervals of minimum length. In order to remove a minimum length interval, it is sufficient to have it neighbor an interval whose length is at least two larger. By placing an offspring in between these two intervals such that it reduces the size of the larger interval, the size of the minimum length interval increases by one if the parent is removed. Lemma 10 bounds the expected number of iterations it takes to remove one minimum length interval like this. In a nutshell, the analysis relies on the fact that minimum length intervals tend to move to the borders of the objective space $[0..n]$. Once there, the minimum distance between a minimum length interval and one with a length at least two larger does not increase. This allows eventually to decrease the number of intervals of minimum length.

Combining everything. We conclude by estimating the time to remove all minimum length intervals of a certain size, and by afterward considering all feasible sizes. We note that our analysis views the progress of the SPEA2 toward an optimal spread by only considering the intervals of minimum length, which is likely a bit pessimistic, as it disregards progress made with other intervals of sub-optimal length.

4.2 The SPEA2 Computes an Optimal Spread Efficiently

We provide the formal details for our outline given in Section 4.1, following the same structure.

No duplicate objective values. We first show that the SPEA2 never reduces the amount of Pareto optima it finds. This is a simple and desirable general property.

Lemma 2. *Consider the SPEA2 optimizing a multi-objective function f, with any parent and offspring population size and with any mutation operator. Let $t \in \mathbb{N}$ such that P_t only contains Pareto-optimal individuals. Then $|f(P_t)| \leq |f(P_{t+1})|$.*

For OMM, this leads immediately to the following bound of objective-function evaluations until each individual has a unique objective value.

Theorem 3. *Consider the SPEA2 optimizing* OMM *with either 1-bit or standard bit mutation and any parent and offspring population size. The expected number of* OMM *evaluations until the parent population contains only distinct objective values or all objective values is $O\big(\mu + \lambda \frac{n \log n}{1-(1-1/\mu)^\lambda}\big)$.*

Moreover, we prove that the extreme solutions of OMM are found quickly.

Theorem 4. *Consider the SPEA2 optimizing* OMM *with either 1-bit or standard bit mutation and any parent and offspring population size. The expected number of* OMM *evaluations until the parent population contains the extreme objective values is $O\big(\mu + \lambda \frac{n \log n}{1-(1-1/\mu)^\lambda}\big)$.*

For 1-bit mutation and $\mu \leq \frac{n}{3}$, this is asymptotically tight.

Useful invariants. As discussed in Section 4.1, our invariants mostly consider the lengths of intervals between the objective values in the current population of the SPEA2 with $\mu \leq n$. To this end, for each iteration $t \in \mathbb{N}$ of Algorithm 1, let $(x_i^t)_{i\in[\mu]}$ denote the individuals from P_t (at the beginning of the `while` loop) in increasing order of their first OMM objective, that is, in increasing order of their number of ones. Moreover, we assume $x_1^t = 0^n$ and $x_\mu^t = 1^n$. Then, for all $i \in [\mu - 1]$, we denote the length of interval i in iteration t by

$$L_i^t \coloneqq \mathrm{OMM}_1(x_{i+1}^t) - \mathrm{OMM}_1(x_i^t). \tag{2}$$

In particular, we are interested in the minimum (and the maximum) length induced by P_t, defined as

$$X_t \coloneqq \min\nolimits_{i\in[\mu-1]} L_i^t \quad \text{and} \quad Y_t \coloneqq \max\nolimits_{i\in[\mu-1]} L_i^t, \tag{3}$$

and their quantity, defined as

$$\begin{aligned} N_t &= |\{i \in [\mu-1] \mid L_i^t = X_t\}| \text{ and} \\ M_t &= |\{i \in [\mu-1] \mid L_i^t = Y_t\}|. \end{aligned} \tag{4}$$

From now on, we focus on the steady-state SPEA2. We first show that the minimum interval length never decreases.

Lemma 5. *Consider the steady-state SPEA2 optimizing* OMM *with either 1-bit or standard bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. The sequence $(X_t)_{t\in\mathbb{N}}$ defined in equation* (3) *is non-decreasing.*

The next result shows that the minimum interval length increases during an iteration or the number of such intervals does not decrease and that the analogous statement holds for the maximum interval length, which does not increase.

Lemma 6. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. The sequences $(-X_t, N_t)_{t\in\mathbb{N}}$ and $(Y_t, M_t)_{t\in\mathbb{N}}$ defined by equations* (3) *and* (4) *are each lexicographically non-increasing.*

The following result shows that if we consider 1-bit mutation, either the offspring or its parent are removed once the intervals have a minimum length of at least two.

Lemma 7. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. Consider an iteration $t \in \mathbb{N}$ such that $X_t > 1$. Last, assume that mutation mutates x into y. Then during the removal phase, we either remove x or y.*

With the next result, we show that intervals of minimum length at the borders of the interval $[0..n]$ do not move away from there if neither the minimum length nor the number of these intervals changes.

Lemma 8. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. Recall equations* (1) *to* (4). *Consider an iteration $t \in \mathbb{N}$ such that $X_t > 1$, that $(-X_t, N_t) > (-\alpha, \beta)$, and that $(X_t, N_t) = (X_{t+1}, N_{t+1})$. Then, if $L_1^t = X_t$, it follows that $L_1^{t+1} = X_t$. Similarly, if $L_{\mu-1}^t = X_t$, then $L_{\mu-1}^{t+1} = X_t$.*

Last for this part, we show that the SPEA2 quickly reaches a state in which the minimum interval length is at least two.

Lemma 9. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. Recall equation* (3), *and let $t \in \mathbb{N}$. If $X_t = 1$, then in an expected number p of $O(\mu^2 n)$ iterations, we have $X_t < X_{t+p}$ and/or $Y_{t+p} \leq 3$.*

Reducing the number of intervals of minimum length. We show that the minimum interval length is quickly increased as long as the SPEA2 did not compute an optimal spread yet. As outlined in Section 4.1, our proof relies on the fact that if an interval of minimum length and one whose length is at least two larger are next to each other, they can be combined into two new intervals whose length is each larger than the current minimum length. To this end, it is important that such two intervals move to each other. We show that this happens via a case distinction with respect to whether a minimum length interval is at either border of $[0..n]$. A crucial building block is Lemma 8, which guarantees that intervals of minimum length remain at the border of $[0..n]$, once they are there. We then show that once we have intervals of minimum length at both borders, then the minimum distance between a minimum length interval and one whose length is at least two larger does not increase. Consequently, these intervals eventually move closer to each other until they create two new intervals of a length larger than the minimum.

Our main result following from the discussion above is the following bound on the number of iterations to increase the minimum interval length or the number of intervals thereof.

Lemma 10. *Consider the steady-state SPEA2 optimizing* OMM *with 1-bit mutation, $\mu \in [n]$, and the initial population containing 0^n and 1^n. Recall equations* (1) *and* (3)*, and let $t \in \mathbb{N}$. If $X_t > 1$ and $(-X_t, N_t) > (-\alpha, \beta)$, then in an expected number p of $O\big(\frac{\mu n \log(\mu)}{X_t}\big)$ iterations, we have $(-X_t, N_t) > (-X_{t+p}, N_{t+p})$.*

Combining everything. We combine all of our prior arguments in order to prove Theorem 1. We essentially rely on Lemma 10 to gradually increase the minimum interval length until it reaches the optimal value α (equation (1)). However, this requires that the minimum interval length is at least 2. By Lemma 9, we achieve such a state quickly, but it requires us to restrict the population size μ of the SPEA2 to at most $\frac{n}{3}$.

Proof of Theorem 1. We first wait until the population does not contain any duplicate objective values, which takes at most $O(\mu n \log n)$ iterations by Theorem 3, and contains the individuals 0^n and 1^n, which takes the same asymptotic amount of time by Theorem 4. Note that this means that the minimum interval length is at least 1. Afterward, we first wait for the first following iteration t^* such that $X_{t^*} > 1$ or $Y_{t^*} \leq 3$. By Lemma 9, we have $t^* = O(\mu^2 n)$.

If $Y_{t^*} \leq 3$, then, recalling equation (1), since $\alpha \geq 3$, then $X_{t^*} = 3$. Thus, $X_{t^*} = \alpha$ and $N_{t^*} = \beta$, as we have $n = \alpha\beta + (\alpha + 1)(\mu - 1 - \beta)$. This concludes this case.

It is left to consider the case $X_{t^*} > 1$. Let $t \in \mathbb{N}_{\geq t^*}$ such that $(-X_t, N_t) > (-\alpha, \beta)$, that is, we did not compute an optimal spread yet. Then by Lemma 10, the expected number of iterations p to achieve either $X_t < X_{t+p}$ or $(-X_{t+p}, N_{t+p}) = (-\alpha, \beta)$ is $O\big(\frac{\mu n \log(\mu)}{X_t}\big)$. Increasing X_t requires reducing N_t at most μ times. Afterward, we sum over all possible values values of X_t, from 1 to $\alpha \leq n$, resulting in the harmonic series and thus concluding the proof. □

5 The NSGA-II Does Not Compute an Optimal Spread in Polynomial Time

We show that the steady-state NSGA-II takes with high probability a super-polynomial time to compute an optimal spread when starting in an unfavorable state. This is in stark contrast to the SPEA2, which always computes an optimal spread in a rather fast polynomial time (Theorem 1). We note that situations similar to the one we discuss seem to be common, as evidenced in [Zheng and Doerr, 2024a, Figure 1].

Theorem 11. *Let $c \in \mathbb{N}_{\geq 2}$ be a constant in n, and assume that $16c$ divides n, and let $n' := \frac{n}{c}$. Consider that the steady-state NSGA-II optimizes* OMM *with 1-bit mutation and with $N = n' + 1$. Apply the definition of equation* (2) *to the NSGA-II. Assume that for all $i \in [N-1] \setminus \{\frac{1}{8}n', \frac{1}{4}n'\}$, we have $L_i^0 = c$, and that we have $L_{n'/8}^0 = c + 1$ and $L_{n'/4}^0 = c - 1$. Then, with probability at least $1 - e^{-\Omega(n)}$, the steady-state NSGA-II does not compute an optimal spread within polynomial time.*

Theorem 11 shows that although the steady-state NSGA-II has computed almost an optimal spread, with only two intervals not being optimal, the algorithm still takes with overwhelming probability a super-polynomial amount of time to compute an optimal spread. This behavior is due to the fact that the removal of an individual in each iteration of the NSGA-II only takes into account the two closest neighbors (in objective values). Thus, empty intervals (of objective values) that neighbor empty intervals of a length that differs by one from their own swap their lengths at random with each other. Due to the shape of the search space of OMM featuring fewer individuals with objective values close to the extreme ones than in the center, this introduces a bias toward individuals with an equal number of zeros and ones during mutation, making it more likely to produce offspring closer to the center. This introduces a bias into how neighboring intervals swap their lengths.

Our counterexamples feature intervals that require the offspring to be placed closer to the extreme objective values than to the center. This implies that the intervals are more likely to drift apart from each other rather than getting closer. Ultimately, this results in an at least super-polynomial runtime with overwhelming probability until intervals of different lengths meet, which is required for an optimal spread.

The SPEA2 does not have this problem as the σ-distances take into consideration more than just the distance to the two closest neighbors in objective space. It is this additional information that allows intervals whose lengths differ by at least two to get closer to each other and align their lengths better (which is the essence of the proof of Lemma 10).

In order to prove Theorem 11, we make use of the following theorem, which shows that processes that move, within a certain interval, in expectation away from a target state only reach their target with a probability exponentially unlikely in the length of the distance to cover.

Theorem 12 (Negative drift [Kötzing, 2016; Krejca, 2019])**.** *Let $(X_t)_{t\in\mathbb{N}}$ be random variables over $\mathbb{R}$. Moreover, let $X_0 \leq 0$, let $b \in \mathbb{R}_{>0}$, and let $T = \inf\{t \in \mathbb{N} \mid X_t \geq b\}$. Suppose that there are values $a \in \mathbb{R}_{\leq 0}$, $\gamma \in (0, b)$, and $\varepsilon \in \mathbb{R}_{<0}$ such that for all $t \in \mathbb{N}$, we have*

(i) $\mathbb{E}[(X_{t+1} - X_t) \cdot \mathbf{1}\{X_t \geq a, t < T\} \mid X_t] \leq \varepsilon \cdot \mathbf{1}\{X_t \geq a, t < T\}$*, that*

(ii) $|X_t - X_{t+1}| \cdot \mathbf{1}\{X_t \geq a, t < T\} < \gamma \cdot \mathbf{1}\{X_t \geq a, t < T\} + \mathbf{1}\{X_t < a, t < T\}$*, and that*

(iii) $X_{t+1} \cdot \mathbf{1}\{X_t \geq a, t < T\} \leq 0$.

Then, for all $t \in \mathbb{N}$, we have $\Pr[T \leq t] \leq t^2 \exp\Big(-\frac{b|\varepsilon|}{2\gamma^2}\Big)$.

Proof of Theorem 11. Note that the only possible changes to the population are those that change where the intervals of length $c+1$ or $c-1$ are located as well as the change that averages the two intervals of lengths $c+1$ and $c-1$ to c. In the latter case, the algorithm computes an optimal spread. However, to do so, it is necessary that those two intervals are next to each other. Before this is the case, they can only move their interval index by one by swapping their position with a neighboring interval of length c.

Let T be the first iteration such that the two intervals of lengths $c+1$ and $c-1$ are neighbors. We show that the probability that T is polynomial is at most $e^{-\Omega(n)}$, thus proving the claim. Hence, in the following, we always assume implicitly that we only consider iterations $t \in T$ such that $t < T$.[2]

For all $t \in \mathbb{N}$, let X_t denote the index of the interval of length $c+1$ at the beginning of iteration t. (Note that we omit the case if no such interval exists, as we only consider iterations less than T.) Analogously, let Y_t denote the index of the interval of length $c-1$ at the beginning of iteration t. Note that both X and Y change by at most 1 in each iteration.

We first consider X and determine its expected change, aiming to apply Theorem 12. To this end, we only consider those iterations in which X actually changes (and iteration 0) and only while X is in $[\frac{1}{16}n'..\frac{3}{16}n']$. Moreover, we assume that the interval of length $c-1$ does not move. Let the resulting process be X', and let T_x be the first point in time $t \in \mathbb{N}$ of process X' such that $X' \geq \frac{3}{16}n'$. Note that $X'_0 = \frac{2}{16}n'$ by assumption. Furthermore note that for all $t \in \mathbb{N}$, if $X'_t < \frac{3}{16}n'$, the same is true for X. Hence, if $Y \geq \frac{3}{16}n'$ until at least T_x, then not considering the interval of length $c-1$ moving does not affect the statements about X' at all. We make use of this observation at the end of this proof.

Since X' changes in each iteration and it always neighbors an interval of length c during this time (by assumption), there are only two cases to consider. Let $t \in \mathbb{N}$. First, X'_t decreases if the steady-state NSGA-II produces an offspring with $c(X'_t - 1) + 1$ ones (from the parent with $c(X'_t - 1)$ ones) and the offspring survives the selection (out of the parent and the offspring). This amounts to a probability of $\big((n - c(X'_t - 1))/n\big) \cdot \frac{1}{2}$. Note that X'_t can decrease since we omit the case that it is the first interval. Second, X'_t increases if an offspring with cX'_t ones is created (from the parent with $cX'_t + 1$ ones) and the offspring is selected. This probability is $\big(cX'_t + 1/n\big) \cdot \frac{1}{2}$. Let $d \coloneqq \big((n - c(X'_t - 1))/n\big) \cdot \frac{1}{2} + \big((cX'_t + 1)/n\big) \cdot \frac{1}{2} \leq 1$. Since X' is conditional on X changing and since we only consider $X'_t \leq \frac{3}{16}n'$, we get

$$\begin{aligned}\mathbb{E}[X'_{t+1} - X'_t \mid X'_t] &= -1 \cdot \frac{n - c(X'_t - 1)}{2nd} + 1 \cdot \frac{cX'_t + 1}{2nd} \\ &= \frac{2cX'_t + c + 1 - n}{2nd} \leq -\frac{1}{2d}\left(1 - \frac{3}{8} - \frac{c+1}{n}\right) \\ &\leq -\frac{1}{2d} \leq -\frac{1}{2}.\end{aligned}$$

We apply Theorem 12 to $(X'_t - \frac{2}{16}n')_{t\in\mathbb{N}}$ with $a = -\frac{1}{16}n'$, $b = \frac{1}{16}n'$, $\gamma = 1$, and $\varepsilon = -\frac{1}{2}$. We get for all $t \in \mathbb{N}$ that

$$\Pr[T_x \leq t] \leq t^2 \exp\Big(-\frac{(1/16)n'(1/2)}{2}\Big) = t^2 e^{-\Omega(n)}.$$

[2]Formally, this requires multiplying each statement with the indicator variable of the event $\{t < T\}$, which we omit.

For Y we proceed analogously, defining Y' in the same way as X' but only consider iterations where Y is in $[\frac{3}{16}n'..\frac{5}{16}n']$. Moreover, let T_y be the first point in time $t \in \mathbb{N}$ of process Y' such that $Y'_t \leq \frac{3}{16}n'$. Last, let $t \in \mathbb{N}$ (less than T_y) and let $d' \coloneqq \big((n - cY'_t)/n\big) \cdot \frac{1}{2} + \big((c(Y'_t - 1) + 1)/n\big) \cdot \frac{1}{2} \leq \frac{1}{2}$. For similar reasons as before, we get

$$\begin{aligned}\mathbb{E}[Y'_t - Y'_{t+1} \mid Y'_t] &= -1 \cdot \frac{n - cY'_t}{2nd'} + 1 \cdot \frac{c(Y'_t - 1) + 1}{2nd'} \\ &= \frac{2cY'_t - c + 1 - n}{2nd'} \leq -\frac{1}{2d'}\left(1 - \frac{3}{8} - \frac{c-1}{n}\right) \\ &\leq -\frac{1}{4d'} \leq -\frac{1}{2}.\end{aligned}$$

By Theorem 12 applied to $(\frac{4}{16}n' - Y'_t)_{t\in\mathbb{N}}$, we get for all $t \in \mathbb{N}$ that $\Pr[T_y \leq t] \leq t^2 \exp(-\frac{(1/16)n'(1/2)}{2}) = t^2 e^{-\Omega(n)}$.

Overall, we see that $X' \geq \frac{3}{16}n'$ and $Y' \leq \frac{3}{16}n'$, which is a necessary condition for the steady-state NSGA-II to compute an optimal spread, occurs for any polynomial number of iterations only with probability $e^{-\Omega(n)}$. Note that since we consider the intersection of the events that $X' < \frac{3}{16}n'$ and $Y' > \frac{3}{16}n'$ for all iterations less than, respectively, T_x and T_y, it does not matter that we did not consider the other interval of length not equal to c for the analysis of either process. Ultimately, since the original process accounts for even more iterations than those considered by X' and Y', the result follows. $\square$

6 Conclusion

We showed rigorously that the steady-state variant of the SPEA2, which generates one offspring each iteration, efficiently computes a desired optimal spread of individuals on the Pareto front of the classic OneMinMax benchmark when the population size is smaller than the size of the Pareto front (Theorem 1). Moreover, we proved in the same setting that the steady-state variant of the NSGA-II can be initialized such that, with overwhelming probability, it does not compute an optimal spread within polynomial time (Theorem 11). This difference in runtime performance is due to the σ-distances used in the SPEA2 for choosing which individual to remove, in contrast to the crowding distance used by the NSGA-II, noting that the σ-distances are computationally more expensive to compute than the crowding distance.

Our current rigorous upper bound for the expected time of the SPEA2 to compute an optimal spread is not tight and also differs from the expected time it takes the algorithm to find the extreme solutions of the Pareto front. This may be due to our proof method, which operates in certain progress levels, only accounting for progress in the worst level, ignoring potential progress elsewhere. Improving this method or showing that the result is actually tight is an interesting open problem.

Another related problem is to prove a similar expected runtime for the SPEA2 when using standard bit mutation instead of 1-bit mutation. The former allows to change individuals more drastically, making it more challenging to measure the progress that is being made within a single iteration.

Acknowledgments

This research benefited from the support of the FMJH Program PGMO.

This work has profited from many deep discussions at the Dagstuhl Seminars 23361 *Multiobjective Optimization on a Budget* and 24271 *Theory of Randomized Optimization Heuristics*.

References

[Auger and Doerr, 2011] Anne Auger and Benjamin Doerr, editors. *Theory of Randomized Search Heuristics*. World Scientific Publishing, 2011.

[Bian and Qian, 2022] Chao Bian and Chao Qian. Better running time of the non-dominated sorting genetic algorithm II (NSGA-II) by using stochastic tournament selection. In *Parallel Problem Solving From Nature, PPSN 2022*, pages 428–441. Springer, 2022.

[Bian *et al.*, 2023] Chao Bian, Yawen Zhou, Miqing Li, and Chao Qian. Stochastic population update can provably be helpful in multi-objective evolutionary algorithms. In *International Joint Conference on Artificial Intelligence, IJCAI 2023*, pages 5513–5521. ijcai.org, 2023.

[Cerf *et al.*, 2023] Sacha Cerf, Benjamin Doerr, Benjamin Hebras, Jakob Kahane, and Simon Wietheger. The first proven performance guarantees for the non-dominated sorting genetic algorithm II (NSGA-II) on a combinatorial optimization problem. In *International Joint Conference on Artificial Intelligence, IJCAI 2023*, pages 5522–5530. ijcai.org, 2023.

[Coello *et al.*, 2007] Carlos Artemio Coello Coello, Gary B. Lamont, and David A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2nd edition, 2007.

[Dang *et al.*, 2023] Duc-Cuong Dang, Andre Opris, Bahare Salehi, and Dirk Sudholt. Analysing the robustness of NSGA-II under noise. In *Genetic and Evolutionary Computation Conference, GECCO 2023*, pages 642–651. ACM, 2023.

[Dang *et al.*, 2024] Duc-Cuong Dang, Andre Opris, and Dirk Sudholt. Crossover can guarantee exponential speed-ups in evolutionary multi-objective optimisation. *Artificial Intelligence*, 330:104098, 2024.

[Deb *et al.*, 2002] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[Deng *et al.*, 2024] Renzhong Deng, Weijie Zheng, Mingfeng Li, Jie Liu, and Benjamin Doerr. Runtime analysis for state-of-the-art multi-objective evolutionary algorithms on the subset selection problem. In *Parallel Problem Solving from Nature, PPSN 2024, Part III*, pages 264–279. Springer, 2024.

[Deng *et al.*, 2025] Renzhong Deng, Weijie Zheng, and Benjamin Doerr. The first theoretical approximation guarantees for the non-dominated sorting genetic algorithm III (NSGA-III). In *International Joint Conference on Artificial Intelligence, IJCAI 2025*. ijcai.org, 2025. Accepted for publication.

[Doerr and Neumann, 2020] Benjamin Doerr and Frank Neumann, editors. *Theory of Evolutionary Computation—Recent Developments in Discrete Optimization*. Springer, 2020. Also available at http://www.lix.polytechnique.fr/Labo/Benjamin.Doerr/doerr_neumann_book.html.

[Doerr and Qu, 2023a] Benjamin Doerr and Zhongdi Qu. A first runtime analysis of the NSGA-II on a multimodal problem. *IEEE Transactions on Evolutionary Computation*, 27:1288–1297, 2023.

[Doerr and Qu, 2023b] Benjamin Doerr and Zhongdi Qu. From understanding the population dynamics of the NSGA-II to the first proven lower bounds. In *Conference on Artificial Intelligence, AAAI 2023*, pages 12408–12416. AAAI Press, 2023.

[Doerr and Qu, 2023c] Benjamin Doerr and Zhongdi Qu. Runtime analysis for the NSGA-II: provable speed-ups from crossover. In *Conference on Artificial Intelligence, AAAI 2023*, pages 12399–12407. AAAI Press, 2023.

[Doerr *et al.*, 2024] Benjamin Doerr, Dimitri Korkotashvili, and Martin S. Krejca. Difficulties of the NSGA-II with the many-objective LeadingOnes problem. *CoRR*, abs/2411.10017, 2024.

[Doerr *et al.*, 2025] Benjamin Doerr, Tudor Ivan, and Martin S. Krejca. Speeding up the NSGA-II with a simple tie-breaking rule. In *Conference on Artificial Intelligence, AAAI 2025*, pages 26964–26972. AAAI Press, 2025.

[Giel and Lehre, 2010] Oliver Giel and Per Kristian Lehre. On the effect of populations in evolutionary multi-objective optimisation. *Evolutionary Computation*, 18:335–356, 2010.

[Giel, 2003] Oliver Giel. Expected runtimes of a simple multi-objective evolutionary algorithm. In *Congress on Evolutionary Computation, CEC 2003*, pages 1918–1925. IEEE, 2003.

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

[Kötzing, 2016] Timo Kötzing. Concentration of first hitting times under additive drift. *Algorithmica*, 75:490–506, 2016.

[Krejca, 2019] Martin S. Krejca. *Theoretical Analyses of Univariate Estimation-of-Distribution Algorithms*. PhD thesis, Universität Potsdam, 2019.

[Kukkonen and Deb, 2006] Saku Kukkonen and Kalyanmoy Deb. Improved pruning of non-dominated solutions based on crowding distance for bi-objective optimization problems. In *Conference on Evolutionary Computation, CEC 2006*, pages 1179–1186. IEEE, 2006.

[Laumanns *et al.*, 2002] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10:263–282, 2002.

[Li *et al.*, 2025] Mingfeng Li, Weijie Zheng, and Benjamin Doerr. Scalable speed-ups for the SMS-EMOA from a simple aging strategy. In *International Joint Conference on Artificial Intelligence, IJCAI 2025*. ijcai.org, 2025. Accepted for publication.

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

[Opris *et al.*, 2024] Andre Opris, Duc Cuong Dang, Frank Neumann, and Dirk Sudholt. Runtime analyses of NSGA-III on many-objective problems. In *Genetic and Evolutionary Computation Conference, GECCO 2024*, pages 1596–1604. ACM, 2024.

[Opris, 2025] Andre Opris. A many objective problem where crossover is provably indispensable. In *International Joint Conference on Artificial Intelligence, IJCAI 2025*. ijcai.org, 2025. To appear. Also available at https://arxiv.org/abs/2412.18375.

[Ren *et al.*, 2024] Shengjie Ren, Chao Bian, Miqing Li, and Chao Qian. A first running time analysis of the Strength Pareto Evolutionary Algorithm 2 (SPEA2). In *Parallel Problem Solving from Nature, PPSN 2024, Part III*, pages 295–312. Springer, 2024.

[Thierens, 2003] Dirk Thierens. Convergence time analysis for the multi-objective counting ones problem. In *Evolutionary Multi-Criterion Optimization, EMO 2003*, pages 355–364. Springer, 2003.

[Wietheger and Doerr, 2023] Simon Wietheger and Benjamin Doerr. A mathematical runtime analysis of the non-dominated sorting genetic algorithm III (NSGA-III). In *International Joint Conference on Artificial Intelligence, IJCAI 2023*, pages 5657–5665. ijcai.org, 2023.

[Wietheger and Doerr, 2024] Simon Wietheger and Benjamin Doerr. Near-tight runtime guarantees for many-objective evolutionary algorithms. In *Parallel Problem Solving from Nature, PPSN 2024, Part IV*, pages 153–168. Springer, 2024.

[Zheng and Doerr, 2023] Weijie Zheng and Benjamin Doerr. Mathematical runtime analysis for the non-dominated sorting genetic algorithm II (NSGA-II). *Artificial Intelligence*, 325:104016, 2023.

[Zheng and Doerr, 2024a] Weijie Zheng and Benjamin Doerr. Approximation guarantees for the non-dominated sorting genetic algorithm II (NSGA-II). *IEEE Transactions on Evolutionary Computation*, 2024. In press, https://doi.org/10.1109/TEVC.2024.3402996.

[Zheng and Doerr, 2024b] Weijie Zheng and Benjamin Doerr. Runtime analysis for the NSGA-II: proving, quantifying, and explaining the inefficiency for many objectives. *IEEE Transactions on Evolutionary Computation*, 28:1442–1454, 2024.

[Zheng and Doerr, 2024c] Weijie Zheng and Benjamin Doerr. Runtime analysis of the SMS-EMOA for many-objective optimization. In *Conference on Artificial Intelligence, AAAI 2024*, pages 20874–20882. AAAI Press, 2024.

[Zheng *et al.*, 2022] Weijie Zheng, Yufei Liu, and Benjamin Doerr. A first mathematical runtime analysis of the non-dominated sorting genetic algorithm II (NSGA-II). In *Conference on Artificial Intelligence, AAAI 2022*, pages 10408–10416. AAAI Press, 2022.

[Zhou *et al.*, 2011] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1:32–49, 2011.

[Zhou *et al.*, 2019] Zhi-Hua Zhou, Yang Yu, and Chao Qian. *Evolutionary Learning: Advances in Theories and Algorithms*. Springer, 2019.

[Zitzler *et al.*, 2001] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK report*, 103, 2001.