Improved Runtime Guarantees for the SPEA2 Multi-Objective Optimizer

Benjamin Doerr[1], **Martin S. Krejca**[1], **Milan Stanković**[2]

[1]Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institut Polytechnique de Paris
[2]École Polytechnique, Institut Polytechnique de Paris
{first-name.last-name}@polytechnique.edu

Abstract

Together with the NSGA-II, the SPEA2 is one of the most widely used domination-based multi-objective evolutionary algorithms. For both algorithms, the known runtime guarantees are linear in the population size; for the NSGA-II, matching lower bounds exist. With a careful study of the more complex selection mechanism of the SPEA2, we show that it has very different population dynamics. From these, we prove runtime guarantees for the ONEMINMAX, LEADINGONESTRAILINGZEROS, and ONEJUMPZEROJUMP benchmarks that depend less on the population size. For example, we show that the SPEA2 with parent population size $\mu \geq n - 2k + 3$ and offspring population size λ computes the Pareto front of the ONEJUMPZEROJUMP benchmark with gap size k in an expected number of $O((\lambda + \mu)n + n^{k+1})$ function evaluations. This shows that the best runtime guarantee of $O(n^{k+1})$ is not only achieved for $\mu = \Theta(n)$ and $\lambda = O(n)$ but for arbitrary $\mu, \lambda = O(n^k)$. Thus, choosing suitable parameters – a key challenge in using heuristic algorithms – is much easier for the SPEA2 than the NSGA-II.

Introduction

Many real-world problems are faced with optimizing several conflicting objectives at the same time. This commonly results in a plethora of incomparable optimal trade-offs (the *Pareto optima*), collectively known as the *Pareto front*. Owing to the incomparable nature of the Pareto optima, it is desirable to have optimization algorithms propose as many of them as possible – ideally the entire front – in as little time as possible. Multi-objective evolutionary algorithms (MOEAs) lend themselves naturally to this kind of task, as they iteratively maintain a set of best-so-far solutions, and they are known to be among the best and most popular approaches for solving multi-objective optimization problems (Coello, Lamont, and van Veldhuizen 2007; Zhou et al. 2011).

Over the last years, the strong performance of modern MOEAs in applications has been steadily complemented by theoretical results, see, e.g., (Li et al. 2016; Zheng, Liu, and Doerr 2022; Do et al. 2023; Wietheger and Doerr 2023; Bian et al. 2023; Zheng and Doerr 2024b; Dang, Opris, and Sudholt 2024; Opris 2025). These works provide rigorous bounds on the expected runtime of MOEAs until they find the entire Pareto front for the first time. Both the results and the arguments used in their proofs provide deeper insights into the working principles – see (Doerr and Qu 2023b) for an analysis of the population dynamics of the NSGA-II –, allow to understand their different advantages and shortcomings – see (Alghouass et al. 2025) for a comparison of the approximation ability of NSGA-II and SPEA2 –, and can help design superior algorithm variants, e.g., (Bian et al. 2023).

In this work, we continue the mathematical study of the *strength Pareto evolutionary algorithm* 2 (SPEA2) (Zitzler, Laumanns, and Thiele 2001). Together with the NSGA-II (Deb et al. 2002), NSGA-III (Deb and Jain 2014), and SMS-EMOA (Beume, Naujoks, and Emmerich 2007), it is one of the widely used domination-based MOEAs. Due to its complex selection mechanism, it is the one least understood from the theoretical perspective. In the first runtime analysis of the SPEA2, Ren et al. (2024a) show that with parent population size μ at least the size of the Pareto front and offspring population size λ at most $O(\mu)$, the SPEA2 computes the Pareto front of the (bi-objective) ONEMINMAX, LEADINGONESTRAILINGZEROS, and ONEJUMPZEROJUMP benchmarks in an expected number of $O(\mu n \log n)$, $O(\mu n^2)$, and $O(\mu n^k)$ function evaluation, respectively. These bounds essentially agree with the runtime guarantees given earlier for the NSGA-II (Zheng and Doerr 2023; Doerr and Qu 2023a) and SMS-EMOA (Bian et al. 2023; Zheng and Doerr 2024b).[1] The only other runtime analysis of the SPEA2 (Alghouass et al. 2025) compares the NSGA-II and SPEA2 in terms of their approximation ability, that is, how well they compute approximations to the Pareto front when the population size smaller than the size of the Pareto front (and only approximations can be computed).

Our contribution: We take a closer look at the more complex selection mechanisms of the SPEA2, which – different from the NSGA-II – takes into account the distances to *all* other individuals on the Pareto front rather than only the two closest ones in each objective. We show that this leads to substantially different population dynamics. While in the NSGA-II, large populations have a tendency to concentrate in the middle of the Pareto front (Doerr and Qu

[1]We note that Ren et al. (2024a) actually showed bounds for arbitrary numbers m of objectives. These were slightly improved in (Wietheger and Doerr 2024) when m is large. Since we only regard the bi-objective case, we do not discuss these results in detail.

2023b), the selection of the SPEA2 distributes multiple objective values evenly on the Pareto front (Lemma 1) for arbitrary multi-objective optimization problems.

This selection property combined with the fact that the SPEA2 creates copies of individuals with moderate probability allows us to prove that in relatively short time, a given objective value is present in the population with the fair multiplicity (Lemma 3). We prove this result for binary representations and bit-wise mutation, but it is clear that comparable results hold for other setting as long as copies of individuals are created sufficiently easily, and that our results also hold when the SPEA2 applies crossover in each iteration with an at most constant probability different from 1.

Knowing that the multiplicities of objective values in the population are evenly distributed, we prove runtime guarantees that suffer less from larger population sizes. Recall from above that previous results for the SPEA2 (and the same holds for all known results of the standard NSGA-II and SMS-EMOA) depend linearly on the population size. Hence, the best performance guarantee for the SPEA2 is obtained from a population size equal to the size of the Pareto front; any increase of the population size increases (that is, weakens) the performance guarantee in a linear fashion. In the bounds we prove in this work, this linear increase kicks in only for larger population sizes. This suggest that in a practical application where the size of the Pareto front is not known in advance, the choice of the population size of the SPEA2 is less critical than of the NSGA-II or SMS-EMOA[2].

We note that the selection problem of the NSGA-II was recently (Doerr, Ivan, and Krejca 2025) overcome by adding a third selection criterion (after non-dominated sorting and crowding distance). This led to runtime guarantees comparable to the ones we prove here. Given that it is a purely theoretical work, there is no information yet on how the modification proposed in (Doerr, Ivan, and Krejca 2025) influences the performance of the NSGA-II in practice. As observed in (Doerr, Ivan, and Krejca 2025), it led to a moderate increase of the time to execute one iteration of the algorithm. In this perspective, it is interesting that our results shows that the standard SPEA2 performs a balanced selection (leading to improved runtime guarantees) without any modification.

Our runtime results. We regard the three common benchmarks ONEMINMAX, LEADINGONESTRAILINGZEROS, and ONEJUMPZEROJUMP of problem size n. Our target is that the population witnesses the Pareto front. Consequently, we always assume that μ is at least the size of the Pareto front, which is at most $n+1$ for our benchmarks. Different from (Ren et al. 2024b), we allow arbitrary offspring population sizes λ, that is, we omit their assumption $\lambda = O(\mu)$.

For ONEMINMAX, one of the easiest benchmarks, we prove that after an expected number of $O((\mu+\lambda)n + n^2 \log n)$ function evaluations, the population witnesses the Pareto front (Theorem 4). Compared to the previous best bound $O(\mu n \log n)$ (for $\lambda = O(\mu)$ only), we observe an asymptotic runtime gain as soon as μ does not have the smallest possible value $\mu = \Theta(n)$. As ONEMINMAX has a Pareto front of size $n+1$, this result shows that the SPEA2 for all $n+1 \leq \mu = O(n \log n)$ and all $\lambda = O(n \log n)$ displays an expected runtime of $O(n^2 \log n)$, which is the one observed with most standard MOEAs. Moreover, this bound is strictly better than the expected runtime of $\Theta(\mu n \log n)$ known for the NSGA-II (Zheng and Doerr 2023; Doerr and Qu 2023b) once $\mu = \omega(n)$.

For LEADINGONESTRAILINGZEROS, we prove an expected runtime of $O((\mu+\lambda)n \log \frac{\mu}{n+1} + n^3 + \lambda n)$ function evaluations (Theorem 12), again when μ is at least the size $n+1$ of the Pareto front. This is $O(n^3)$, the typical runtime of standard MOEAs on this benchmark, for all $n+1 \leq \mu = O(n^2)$ and all $\lambda = O(n^2)$.

For ONEJUMPZEROJUMP problems with gap size $k \in [2, \frac{n}{2}] \cap \mathbb{Z}$, the improvement is even more drastic. We prove an expected runtime of $O((\mu+\lambda)n + n^{k+1})$ function evaluations (Theorem 7), for $\mu \geq n - 2k + 3$. That is, the typical runtime of $O(n^{k+1})$ is achieved for the large range of all $n - 2k + 3 \leq \mu = O(n^k)$ and all $\lambda = O(n^k)$.

In summary, our results show that the SPEA2 attains the runtime which many known MOEAs achieve only with asymptotically optimal parameter values for much larger ranges of values. While naturally the precise bounds we prove depend on the optimization problem, they key arguments leading to these bounds, the balanced selection of the SPEA2 and the multiplicative growth of the number of individuals having a certain objective value, hold under very general assumptions. For this reason, it appears justified to claim that the SPEA2 is much more robust to changes of its parameters than the main other domination-based MOEAs.

Full version. The full version of this paper is available on arXiv (Doerr, Krejca, and Stanković 2025).

Preliminaries

We denote by $\mathbb{Z}$ the set of integers. For $m, n \in \mathbb{Z}$, $m \leq n$, we define $[m..n] \coloneqq [m, n] \cap \mathbb{Z}$.

For $n \in \mathbb{Z}_{\geq 1}$ and a bit string $x = (x_i)_{i \in [1..n]} \in \{0,1\}^n$, we define $|x|_1 \coloneqq \sum_{i=1}^n x_i$ as the number of ones of x, and $|x|_0 \coloneqq n - |x|_1$ as the number of zeros of x. We define $1^n \coloneqq (1)_{i \in [1..n]}$ as the bit string containing only ones, and $0^n \coloneqq (0)_{i \in [1..n]}$ as the bit string containing only zeros.

Multi-Objective Optimization

Let Ω be some set, and let $m \in \mathbb{Z}_{\geq 2}$. Consider a function

$$f \colon \Omega \to \mathbb{R}^m, x \mapsto (f_1(x), \dots, f_m(x)).$$

We consider *maximizing* f with respect to the *dominance* partial order, defined as follows. Let $x, y \in \Omega$. If for all $i \in [1..m]$ we have $f_i(x) \geq f_i(y)$, then we say x *weakly dominates* y and write $x \succeq y$. Furthermore, if $x \succeq y$ and $f(x) \neq f(y)$, then we say x *strictly dominates* y and write $x \succ y$. In all other cases, we say x and y are *incomparable*.

If a point $x \in \Omega$ is not strictly dominated by any other point in Ω, then x is *Pareto-optimal*. The set $S^* = \{x \in \Omega \mid x \text{ is Pareto-optimal }\}$ of Pareto-optimal points is called the *Pareto set*, and the set of their objective values, $F^* = \{f(x) \mid x \in S^*\}$, is called the *Pareto front*. We aim at finding a subset of the Pareto set that *covers* the Pareto front, that is, finding a set $A \subseteq S^*$ such that $\{f(x) \mid x \in A\} = F^*$.

[2]There are no lower bounds for the SMS-EMOA, but we expect behavior similar to the NSGA-II (Doerr and Qu 2023b).

The SPEA2

Algorithm 1 shows the generic SPEA2 maximizing a multi-objective function $f\colon \Omega \to \mathbb{R}^m$, similar to the pseudocode provided by (Wietheger and Doerr 2024), who use updated notation (Table 1).

We focus on the case where Ω is the set of bit strings of length $n \in \mathbb{Z}_{\geq 1}$, i.e., $\Omega = \{0,1\}^n$, known as *pseudo-Boolean maximization*. We call the sets P_t and Q_t in Algorithm 1 the *parent* and the *offspring population*, respectively. We emphasize that these are multi-sets, i.e, repetitions of elements are allowed. The set operations $\cup$ and $\setminus$ are the multi-set union and the multi-set difference. We also call the multi-set $R_t = P_t \cup Q_t$, i.e., the combined parent and offspring population, the *total population* in iteration $t \in \mathbb{Z}_{\geq 0}$.

In case $\Omega = \{0,1\}^n$, the **mutate** function applies the standard bit mutation to a given bit string. That is, for all $x \in \{0,1\}^n$, a call to $\mathrm{mutate}(x)$ flips each bit of a copy of the bit string x independently with probability $\frac{1}{n}$, and returns the resulting bit string.

The function **eliminate** removes one individual from the given multi-set, according to a concept called *σ-criterion*. The σ-criterion favors individuals that are further away from their neighbors, in the objective space. Formally, let A be a multi-set of elements from Ω, and let $f(A)$ be the multi-set $\{f(x) \mid x \in A\}$. For $x \in A$ and $k \in [1..|A|-1]$, we define $\sigma_k(x)$ as the distance between $f(x)$ and its k-th nearest neighbor in $f(A)$. That is, we sort the list $(\|f(x) - f(y)\|_2)_{y \in A \setminus \{x\}}$ in increasing order, and define $\sigma_k(x)$ as the k-th member of the sorted list. Furthermore, we define the relation $\leq_d$ as the lexicographic order over $\{(\sigma_k(z))_{k \in [1..|A|-1]}\}_{z \in A}$. Hence, for all $x, y \in A$, we have $x \leq_d y$ if and only if one of the following holds.

(i) For all $k \in [1..|A|-1]$, we have $\sigma_k(x) = \sigma_k(y)$.

(ii) There exists a $k \in [1..|A|-1]$ such that $\sigma_k(x) < \sigma_k(y)$, and, for all $\ell \in [1..k-1]$, we have $\sigma_\ell(x) = \sigma_\ell(y)$.

A call to $\mathrm{eliminate}(A)$ removes an element x from A which is minimal with respect to $\leq_d$. If there are multiple such elements, we break the tie uniformly at random.

We note that the σ-criterion initially requires $O(\mu^2)$ computations of shortest distances among all pairs of individuals, and it requires $O(\mu)$ shortest-distance computations afterward for each individual added or removed. This can be expensive when compared to selection criterions from other MOEAs, such as the NSGA-II. However, when considering that a population does not change by that much from one iteration to the next and by keeping track of the number of copies of objective values (which eliminates the need to recompute all known shortest distances), the overall cost of this operation can be reduced. Since we focus on the number of function evaluations, for which a single evaluation is usually already very costly, the exact cost of the σ-criterion is not relevant to our analysis but remains an important aspect to keep in mind when interpreting our results.

The **add_dominated** function returns an element from a given set according to some assignment of objective values. If, in iteration t, the number of non-dominated individuals in the total population R_t is less than the parent population size μ, the algorithm uses this function to select dominated individuals from the total population R_t and include them in the new parent population P_{t+1}. Note that the multi-set $R_t \setminus P_{t+1}$, which the $\mathrm{add\_dominated}$ function is applied to, contains only the individuals that are dominated by another individual from R_t. However, in our analysis, we mainly focus on the process of generating new non-dominated individuals from the existing non-dominated individuals. Hence, the exact formulation of the $\mathrm{add\_dominated}$ function is not essential for the analysis. We refer the reader to the original SPEA2 paper for a detailed description of this method (Zitzler, Laumanns, and Thiele 2001).

Algorithm 1: The *strength Pareto evolutionary algorithm* 2 (Zitzler, Laumanns, and Thiele 2001) (SPEA2) with parent population size $\mu \in \mathbb{Z}_{\geq 1}$ and offspring population size $\lambda \in \mathbb{Z}_{\geq 1}$, using standard bit mutation, maximizing an m-objective function $f\colon \Omega \to \mathbb{R}^m$.

$P_0 \leftarrow$ population of μ individuals sampled independently, uniformly at random from Ω;
for $t \in \mathbb{Z}_{\geq 0}$ **do**
  if termination criterion is met **then**
    return non-dominated individuals from P_t;
  $Q_t \leftarrow \emptyset$;
  for $i \in [1..\lambda]$ **do**
    Select x from P_t uniformly at random;
    $Q_t \leftarrow Q_t \cup \{\mathrm{mutate}(x)\}$;
  $R_t \leftarrow P_t \cup Q_t$;
  $S_t \leftarrow$ non-dominated individuals from R_t;
  while $|S_t| > \mu$ **do**
    $\mathrm{eliminate}(S_t)$;
  while $|S_t| < \mu$ **do**
    $S_t \leftarrow S_t \cup \{\mathrm{add\_dominated}(R_t \setminus S_t)\}$;
  $P_{t+1} \leftarrow S_t$;

Original SPEA2	Our SPEA2 formulation
archive $\overline{P}_t$	parent population P_t
archive size $\overline{N}$	parent population size μ
population P_t	offspring population Q_t
population size N	offspring population size λ

Table 1: Comparison of the original notation in (Zitzler, Laumanns, and Thiele 2001) and our notation.

Original SPEA2 Although Algorithm 1 may seem different from the original SPEA2 by Zitzler, Laumanns, and Thiele (2001), the two algorithms are in fact almost equivalent. The terms *archive* and *population* from the original algorithm correspond respectively to the *parent population* and the *offspring population* in our formulation, which uses a more standard terminology for the field of runtime analysis. Table 1 (Wietheger and Doerr 2024, Table II) summarizes the different notation used in the two formulations.

Using our notation, we compare the two algorithms. The original SPEA2 starts with an empty parent population and a random offspring population, whereas for our variant, it is vice versa. Furthermore, within one iteration of the original SPEA2, the selection for survival is performed before offspring generation, which is not a common order for evolutionary algorithms. Our formulation takes the more common order, that is, offspring generation is performed first. In the original SPEA2, when selection for survival is performed for the first time, there are λ individuals to choose from (empty parent population and random offspring population), meaning that the algorithm is not properly defined for $\lambda < \mu$. In our variant, there are always $\mu + \lambda$ individuals to choose from when selecting for survival, hence it is well defined for all values λ and μ. The last difference between these two algorithms is in the way parents are selected for offspring generation. The original SPEA2 uses binary tournaments, whereas our formulation selects parents uniformly at random. We let the reader check that, apart from the differences mentioned above, the two algorithms are equivalent.

Benchmarks

We analyze the runtime of SPEA2 for the ONEMINMAX, ONEJUMPZEROJUMP, and LEADINGONESTRAILINGZEROS benchmark functions. These three benchmarks are among the most established ones for theoretical runtime analysis of MOEAs, featured in almost all MOEA theory results. All functions are defined on the set of bit strings of length $n \in \mathbb{Z}_{\geq 1}$.

The **ONEMINMAX (OMM)** benchmark, proposed in (Giel and Lehre 2010), is a bi-objective function that returns the number of 0s and 1s in a bit string. Formally, we have

$$\text{OMM}\colon \{0,1\}^n \to \mathbb{R}^2, x \mapsto (|x|_0, |x|_1).$$

In this setting, every individual is Pareto-optimal, and the Pareto front is $\{(k, n-k) \mid k \in [0..n]\}$.

The **ONEJUMPZEROJUMP (OJZJ)** benchmark (Doerr and Zheng 2021) is similar to OMM in a way that both functions depend only on the total number of ones in a bit string. ONEJUMPZEROJUMP, however, has an additional parameter $k \in [2..\lfloor n/2 \rfloor]$, called *gap size*. Only the individuals that have between k and $n-k$ ones, and the individuals 1^n and 0^n are Pareto-optimal, and they dominate all other individuals. Formally, for $n \in \mathbb{Z}_{\geq 2}$, $k \in [2..n]$, the ONEJUMPZEROJUMP benchmark with gap size k (OJZJ$_k$) is defined as

$$\text{OJZJ}_k\colon \{0,1\}^n \to \mathbb{R}^2, x \mapsto (f_1(x), f_0(x)),$$

where, for $i \in \{0,1\}$, we have

$$f_i(x) = \begin{cases} k + |x|_i & \text{if } |x|_i \leq n-k \text{ or } |x|_i = n, \\ n - |x|_i & \text{otherwise.} \end{cases}$$

The Pareto front is $F^* = \{(i, n+2k-i) \mid i \in [2k..n] \cup \{k, n+k\}\}$, and the Pareto set is $S^* = \{x \in \{0,1\}^n \mid |x|_1 \in [k..n-k] \cup \{0,n\}\}$ (Doerr and Zheng 2021, Theorem 5). We define $F_I^* \coloneqq \{(i, n+2k-i) \mid i \in [2k..n]\}$ as the inner region of the Pareto front, and $S_I^* \coloneqq \{x \in \{0,1\}^n \mid |x|_1 \in [k..n-k]\}$ as the inner region of the Pareto set. Last, the largest set of pairwise incomparable solutions has a cardinality of at most $n - 2k + 3$ (Doerr and Zheng 2021, Corollary 6).

The **LEADINGONESTRAILINGZEROS (LOTZ)** (Laumanns, Thiele, and Zitzler 2004) is a bi-objective function

$$f\colon \{0,1\}^n \to \mathbb{R}^2, x \mapsto (f_1(x), f_2(x)),$$

where f_1 returns the length of the longest prefix of $1s$, and f_2 returns the longest suffix of $0s$. Formally,

$$f_1(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j \quad \text{and} \quad f_2(x) = \sum_{i=1}^{n} \prod_{j=0}^{i-1} (1 - x_{n-j}).$$

The Pareto front is $\{(k, n-k) \mid k \in [0..n]\}$, which is also the largest set of incomparable values. The Pareto set is $\{1^k 0^{n-k} \mid k \in [0..n]\}$. Contrary to OMM and OJZJ$_k$, in LOTZ, we have a 1-to-1 correspondence between the Pareto set and the Pareto front. Hence, covering the Pareto front is equivalent to finding all Pareto-optimal individuals.

Population Dynamics of the SPEA2

The main reason for the superior runtimes we prove for the SPEA2 in this work are its different population dynamics. As we will show in the following analysis, the selection mechanism of the SPEA2, different from the one of the classic NSGA-II, removes from the set of non-dominated solutions always an individual with objective value most present in this set. This results in different objective values being present in an as balanced manner as possible. In particular, there is a tendency to increase the number of individuals with a recently found solution value, which in turn aides finding an unseen solution value from these.

The following lemma makes our finding on the more balanced selection of the SPEA2 precise.

Lemma 1. *Consider the SPEA2 with $\mu, \lambda \in \mathbb{Z}_{\geq 1}$ optimizing some multi-objective problem $f\colon \Omega \to \mathbb{R}^m$, and consider an arbitrary iteration $t \in \mathbb{Z}_{\geq 0}$. As in Algorithm 1, let S_t be the multi-set of non-dominated individuals from the total population R_t. Let $F_t = \{f(x) \mid x \in S_t\}$ be the set of distinct objective values of individuals from S_t. Let $u \in \mathbb{R}^m$, and let $A_u = \{x \in S_t \mid f(x) = u\}$ be the multi-set of non-dominated individuals with objective value u.*[3]

Then at least $\min\{|A_u|, \lfloor \frac{\mu}{|F_t|} \rfloor\}$ individuals from A_u survive to the next iteration.

In particular, if M is the maximum size of an incomparable set of solutions of f, then at least $\min\{|A_u|, \lfloor \frac{\mu}{M} \rfloor\}$ individuals from A_u survive to the next iteration.

Proof. There is nothing to show if $A_u = \emptyset$, so let us assume that S_t contains individuals with objective value u. If $|S_t| \leq \mu$, all non-dominated individuals survive, and the result again follows trivially. Hence, we assume that $|S_t| > \mu$.

In iteration t, the algorithm eliminates $|S_t| - \mu$ individuals from S_t, one by one, according to the σ_k criterion. Let

[3] Clearly, A_u is empty if there is no individual with objective value u, or if such individuals are strictly dominated by others. It is nevertheless convenient to have also these trivial cases covered.

$i \in [0..|S_t| - \mu]$ and assume that i individuals have already been removed. Denote by S_t^i and A_u^i the multi-sets of the remaining individuals from S_t and A_u, respectively. Note that $A_u^0 = A_u$, and $A_u^{|S_t|-\mu}$ represents the individuals from A_u that survive to iteration $t+1$.

Our goal is to show that, if $i \leq |S_t| - \mu - 1$ and $|A_u^i| \leq \lfloor \frac{\mu}{|F_t|} \rfloor$, then we do not remove an individual from A_u^i, i.e, we have $A_u^i = A_u^{i+1}$. With an elementary induction, this implies that $A_u^{|S_t|-\mu} \geq \min\{|A_u|, \lfloor \frac{\mu}{|F_t|} \rfloor\}$, and the result follows.

Assume that $i \leq |S_t| - \mu - 1$ and $|A_u^i| \leq \lfloor \frac{\mu}{|F_t|} \rfloor$. Since $\{A_v^i\}_{v \in F_t}$ is a partition of S_t^i and $|S_t^i| = |S_t| - i > \mu$, by the pigeonhole principle, there exists $v \in F_t$ such that $|A_v^i| > \lfloor \frac{\mu}{|F_t|} \rfloor \geq |A_u^i|$.

Now, we show that no element $y \in A_u^i$ is eliminated. Let $x \in A_v^i$ and $y \in A_u^i$. Then, since $|A_v^i| \geq |A_u^i| + 1$, for all $k \in [1..|A_u^i| - 1]$, we have $\sigma_k(x) = \sigma_k(y) = 0$, but $\sigma_{|A_u^i|}(x) = 0 < \sigma_{|A_u^i|}(y)$. Thus, we have $y \not\leq_d x$, meaning that y is not minimal with respect to $\leq_d$. Therefore, y is not removed. Since this holds for any $y \in A_u^i$, the set A_u^i remains unchanged. This proves the main claim.

The last claim follows immediately from the fact $|F_t| \leq \overline{M}$. So see the latter, take any collection S of individuals such that S contains exactly one individual for each objective value in F_t. Then S is an incomparable set and hence $|F_t| = |S| \leq \overline{M}$. □

In the remainder, we restrict ourselves to the common case of bit string spaces, that is, $\Omega = \{0,1\}^n$. We note that comparable results could be shown for other search spaces.

We now build on the balanced selection property and show that the number of individuals with a given objective values exhibits a multiplicative growth behavior. In this analysis, we use the following bound on the expected hitting time of a stochastic process that exhibits a certain kind of stochastic multiplicative growth. We note that the non-intuitive constant 0.29 stems from the fact that we use an estimate on binomial distributions from (Doerr 2018).

Lemma 2. *Let $k \in \mathbb{Z}_{\geq 1}$ and $r \in (0,1)$ such that $kr \geq 0.29$. Consider two sequences of integer-valued random variables $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ and $(Y_t)_{t \in \mathbb{Z}_{\geq 1}}$ such that, for all $t \in \mathbb{Z}_{\geq 0}$, we have*

$$X_0 = 1,$$
$$Y_{t+1} \succeq \mathrm{Bin}(k, \min\{rX_t, 1\}), \text{ and}$$
$$X_{t+1} = X_t + Y_{t+1}.$$

Let $B \in [1, 1/r]$ and $T = \inf\{t \geq 0 \mid X_t \geq B\}$. Then

$$\mathrm{E}[T] \leq 4\lceil \log_{1+kr} B \rceil.$$

We now conduct the promised analysis of how the number of individuals with a given objective value grows.

Lemma 3. *Consider an iteration $t_0 \in \mathbb{Z}_{\geq 0}$ of the SPEA2 optimizing a function $f\colon \{0,1\}^n \to \mathbb{R}^m$, where $n \geq 3$. Let $\overline{M}$ be the size of the largest set of pairwise incomparable objective values of f, and assume that the parent population size is $\mu \geq \overline{M}$. Let R_t and F_t be as in Lemma 1. Consider a non-dominated objective value $v \in F_{t_0}$. For $t \in \mathbb{Z}_{\geq 0}$, define $X_t = |\{x \in R_{t_0+t} \mid f(x) = v\}|$. Let $B \in [1..\lfloor \frac{\mu}{\overline{M}} \rfloor]$, and let T be the smallest integer t such that $X_t \geq B$ or v is strictly dominated by a $u \in F_{t_0+t}$. Then $\mathrm{E}[T] = O(\lceil \frac{\mu}{\lambda} \rceil \log B)$.*

Proof. Our goal is to apply Lemma 2 to the process $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ in order to obtain a bound on $\mathrm{E}[T]$. Thus, we carefully analyze the behavior of $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$.

Since $v \in F_{t_0}$, there exists $x \in R_{t_0}$ such that $f(x) = v$, hence $X_0 \geq 1$. Let $t \in [0..T-1]$. Note that, since all values in F_{t_0+t} are non-dominated, they are also pairwise incomparable. Thus, we have $|F_{t_0+t}| \leq \overline{M}$, and $X_t < B \leq \lfloor \frac{\mu}{\overline{M}} \rfloor \leq \lfloor \frac{\mu}{|F_{t_0+t}|} \rfloor$. Since $t < T$, all individuals $x \in R_{t_0+t}$ with $f(x) = v$ are non-dominated. Therefore, by Lemma 1, at least $\min\{B, X_t\} = X_t$ of them are present in P_{t_0+t+1}. All these individuals are also kept in R_{t_0+t+1}. Thus, $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ is non-decreasing.

We now estimate $X_{t+1} - X_t$. In every iteration, the offspring population is generated by λ times choosing a parent uniformly at random and applying standard bit mutation. Recall that there are at least X_t individuals x with $f(x) = v$ in the parent population P_{t_0+t+1}. Hence, when generating offspring in iteration $t_0 + t + 1$, the probability of choosing a parent with $f(x) = v$ is at least $\frac{X_t}{\mu}$. Thus, the probability of choosing such a parent and generating a copy of it as an offspring is at least $\frac{X_t}{\mu}(1 - \frac{1}{n})^n$. Let $\delta = 0.29$. Since $(1 - \frac{1}{n})^n$ is increasing in n and, by assumption, $n \geq 3$, we have $(1 - \frac{1}{n})^n \geq \frac{8}{27} > 0.29$ and, therefore, $\frac{X_t}{\mu}(1 - \frac{1}{n})^n \geq \delta \frac{X_t}{\mu}$. Thus, we have $X_{t+1} - X_t \succeq \mathrm{Bin}(\lambda, \delta \frac{X_t}{\mu})$.

We observe that $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ exhibits multiplicative growth, similar to the one required by Lemma 2. However, we cannot apply the lemma because we cannot guarantee that $\delta \frac{\lambda}{\mu} \geq 0.29$. It is the factor of $\frac{\lambda}{\mu}$ in the previous inequality that is problematic. Thus, we consider the progress that $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ makes in $\lceil \frac{\mu}{\lambda} \rceil$ iterations, that is, we study the subsequence $(X_{t\lceil \frac{\mu}{\lambda} \rceil})_{t \in \mathbb{Z}_{\geq 0}}$ of $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$.

Let $t \in [0..\lfloor T / \lceil \frac{\mu}{\lambda} \rceil \rfloor - 1]$. We estimate $X_{(t+1)\lceil \frac{\mu}{\lambda} \rceil} - X_{t\lceil \frac{\mu}{\lambda} \rceil}$ using similar arguments as before. Let $s \in [t\lceil \frac{\mu}{\lambda} \rceil..(t+1)\lceil \frac{\mu}{\lambda} \rceil - 1]$. The number of individuals x with $f(x) = v$ in the parent population P_{t_0+s+1} is at least $X_s \geq X_{t\lceil \frac{\mu}{\lambda} \rceil}$. Hence, in iteration $t_0 + s + 1$, the probability of choosing a parent x with $f(x) = v$ and generating a copy of it as an offspring is at least $\frac{X_{t\lceil \mu/\lambda \rceil}}{\mu}(1 - \frac{1}{n})^n \geq \delta \frac{X_{t\lceil \mu/\lambda \rceil}}{\mu}$. Since λ offspring are generated per iteration, the total number of offspring generated in iterations $[t\lceil \frac{\mu}{\lambda} \rceil..(t+1)\lceil \frac{\mu}{\lambda} \rceil - 1]$ is $\lambda \lceil \frac{\mu}{\lambda} \rceil$. Therefore, we have $X_{(t+1)\lceil \frac{\mu}{\lambda} \rceil} - X_{t\lceil \frac{\mu}{\lambda} \rceil} \succeq \mathrm{Bin}(\lambda \lceil \frac{\mu}{\lambda} \rceil, \delta \frac{X_{t\lceil \mu/\lambda \rceil}}{\mu})$.

Let $\widetilde{T}$ be the smallest $t \in \mathbb{Z}_{\geq 0}$ such that $X_{t\lceil \frac{\mu}{\lambda} \rceil} \geq B$ or v is strictly dominated by some $u \in F_{t_0+t\lceil \frac{\mu}{\lambda} \rceil}$. We apply Lemma 2 to the sequence $(X_{t\lceil \frac{\mu}{\lambda} \rceil})_{t \in \mathbb{Z}_{\geq 0}}$ to bound $\mathrm{E}[\widetilde{T}]$. Note that the sequence $(X_{t\lceil \frac{\mu}{\lambda} \rceil})_{t \in \mathbb{Z}_{\geq 0}}$ exhibits multiplicative growth required by Lemma 2 before it reaches the bound B or a value strictly better than v is found. That is, for $t \in [0..\widetilde{T} - 2]$ we have $X_{(t+1)\lceil \frac{\mu}{\lambda} \rceil} - X_{t\lceil \frac{\mu}{\lambda} \rceil} \succeq \mathrm{Bin}(\lambda \lceil \frac{\mu}{\lambda} \rceil, \delta \frac{X_{t\lceil \mu/\lambda \rceil}}{\mu})$. For $t \geq \widetilde{T} - 1$ we are not concerned

by the distribution of $X_{(t+1)\lceil \frac{\mu}{\lambda} \rceil} - X_{t\lceil \frac{\mu}{\lambda} \rceil}$ because either $X_{(t+1)\lceil \frac{\mu}{\lambda} \rceil}$ already reaches the bound B, or $F_{t_0+(t+1)\lceil \frac{\mu}{\lambda} \rceil}$ contains a value strictly better than v. We verify the other assumptions of Lemma 2. First, we have $\lambda\lceil \frac{\mu}{\lambda} \rceil \frac{\delta}{\mu} \geq \delta = 0.29$. Second, since $\overline{M} \geq 1 > \delta$, we also have $B \leq \lfloor \frac{\mu}{\overline{M}} \rfloor \leq \frac{\mu}{\delta}$. Thus, by Lemma 2, we have $\mathrm{E}[\widetilde{T}] \leq 4\lceil \log_{1+\lambda\lceil \frac{\mu}{\lambda} \rceil \frac{\delta}{\mu}} B \rceil$. Moreover, since $X_{\widetilde{T}\lceil \mu/\lambda \rceil} \geq B$ or $F_{t_0+\widetilde{T}\lceil \frac{\mu}{\lambda} \rceil}$ contains a value strictly better than v, we have $T \leq \widetilde{T}\lceil \mu/\lambda \rceil$. Therefore, $\mathrm{E}[T] \leq \mathrm{E}[\widetilde{T}]\lceil \mu/\lambda \rceil = O(\lceil \mu/\lambda \rceil \log B)$. □

Runtime Analysis

We now conduct our mathematical runtime analyses of the SPEA2 on the ONEMINMAX, ONEJUMPZEROJUMP, and LOTZ benchmarks. The main work for this has been done in the previous section. However, we still need a couple of new arguments compared to the existing runtime analyses of other MOEAs on these benchmarks.

Runtime Analysis for OneMinMax

For the OMM problem, we prove in Theorem 4 that when the parent population size of the SPEA2 satisfies $\mu \geq n+1$, the expected runtime is $O(\frac{\mu n}{\lambda} + n + \frac{n^2}{\lambda} \log n)$ iterations, or $O(\mu n + \lambda n + n^2 \log n)$ function evaluations. Consequently, when the total population size $\lambda + \mu$ is $O(n \log n)$, the algorithm takes $O(n^2 \log n)$ function evaluations in expectation with no further influence on the population sizes.

Theorem 4. *The expected runtime of the SPEA2 with parent population size $\mu \geq n+1$ and offspring population size λ optimizing the* OMM *benchmark is $O(\frac{\mu n}{\lambda} + n + \frac{n^2}{\lambda} \log n)$ iterations, or $O((\mu+\lambda)n + n^2 \log n)$ function evaluations.*

Our analysis uses an approach similar to the one of Doerr, Ivan, and Krejca (2025) for the balanced NSGA-II, but with two key differences. First, we do not use the multiplicative up-drift theorem (Doerr and Kötzing 2021, Theorem 2.1), as it is not applicable here, but instead use Lemma 2. Second, we extend the bound to potentially different parent and offspring population sizes, which are common for the SPEA2 but not for the (balanced) NSGA-II.

The broad outline of our proof of Theorem 4 is to add up the waiting times for finding a new objective value on the Pareto front from a neighboring value. These times are estimated in Lemma 5, using Lemma 3 to analyze the time until a suitable number of individuals with the neighboring objective value are in the population and then estimating the time it takes to generate from one of these parents a new objective value. To obtain the best estimate for the runtime, the number of these parents we wait for depends on the difficulty of the mutation generating the desired offspring.

Lemma 5. *Let $n \geq 3$. Consider the SPEA2 with parent population size $\mu \geq n+1$ and offspring population size λ, optimizing the* OMM *benchmark. Let $v \in [1..n]$ and $i \in \{0,1\}$. Assume that in some iteration $t_0 \in \mathbb{Z}_{\geq 0}$ there exists an individual $x_0 \in P_{t_0}$ such that $|x_0|_i = v$. Denote by T_v^i the number of iterations needed to generate an individual y with $|y|_i = v-1$. Then*

$$\mathrm{E}[T_v^i] = O\Big(\lceil \tfrac{\mu}{\lambda} \rceil \log \lceil \tfrac{n}{v} \rceil + \tfrac{\mu}{\lambda} + \tfrac{n^2}{\lambda v} + 1\Big).$$

Our main results now follows from adding up the waiting times from the previous lemma in a suitable order, a standard argument in the analysis of OMM, and simplifying this sum. As it will be profitable in the analysis of ONEJUMPZEROJUMP, we show a more general result, discussing the time to cover a subinterval of the Pareto front.

Theorem 6. *Consider the SPEA2 with parent population size $\mu \geq n+1$ and offspring population size λ, optimizing the* OMM *benchmark. Let $\alpha \in [0..\lfloor \frac{n}{2} \rfloor]$, and let N_α denote the number of iterations it takes to cover the subset $\{(v, n-v)\}_{v \in [\alpha..n-\alpha]}$ of the Pareto front, that is, $N_\alpha = \inf\{t \in \mathbb{Z}_{\geq 0} \mid [\alpha..n-\alpha] \subset \{|x|_1 \mid x \in P_t\}\}$. Assume there exists an individual $x_0 \in P_0$ such that $|x|_1 \in [\alpha..n-\alpha]$. Then, $\mathrm{E}[N_\alpha] = O(\frac{\mu n}{\lambda} + n + \frac{n^2}{\lambda} \log n)$.*

Applying Theorem 6 with $\alpha = 0$ (for this we note that, trivially, the random initial population contains individuals on the Pareto front) we obtain the claimed runtime bound for the SPEA2 applied to OMM (Theorem 4).

Runtime Analysis for ONEJUMPZEROJUMP

We move on to the ONEJUMPZEROJUMP benchmark and show in Theorem 7 that for all gap sizes $k \in [2..\lfloor n/2 \rfloor]$, the SPEA2 computes the Pareto front in an expected runtime of $O(\frac{n^{k+1}}{\lambda} + \frac{\mu}{\lambda} n + n)$ iterations, or $O(n^{k+1} + \mu n + \lambda n)$ function evaluations, when the parent population size is at least the size of the Pareto front, that is, $\mu \geq n - 2k + 3$. Consequently, unless a population size exceeds the asymptotic order of n^k, the runtime bound in function evaluations is (asymptotically) independent of the population sizes.

Until the end of the section, we consider the SPEA2 with general population sizes μ and λ, optimizing the OJZJ_k with gap size $k \in [2..\lfloor n/2 \rfloor]$. Note that this implies $n \geq 4$.

Theorem 7. *The expected runtime of the SPEA2 with parent population size $\mu \geq n - 2k + 3$ and offspring population size λ on the OJZJ_k problem with gap size $k \in [2..\lfloor n/2 \rfloor]$ is $O(\frac{n^{k+1}}{\lambda} + \frac{\mu}{\lambda} n + n)$ iterations, or $O(n^{k+1} + \mu n + \lambda n)$ function evaluations.*

In our analysis, we use several times the following elementary estimate to translate success probabilities of single mutations into runtime estimates.

Lemma 8. *Let $\lambda \in \mathbb{Z}_{\geq 1}$. Consider a sequence of independent random Bernoulli trials, each with success probability $q \in (0,1]$. We call λ non-overlapping consecutive trials an* iteration. *Let T denote the first iteration containing a success. Then $\mathrm{E}[T] \leq 1 + \frac{1}{q\lambda}$.*

As in (Doerr and Qu 2023a), we decompose the optimization process into three stages, which we analyze separately. The first stage estimates the time to find any solution on the inner part of the Pareto front. To this end, we do not require yet the main machinery of this work.

Lemma 9. *Let T_1 be the number of iterations needed to find the first inner Pareto optimum, that is $T_1 = \inf\{t \in \mathbb{Z}_{\geq 0} \mid P_t \cap S_I^* \neq \emptyset\}$. Then $\mathrm{E}[T_1] \leq \frac{e}{\lambda} k^k + 1$.*

From now on, we assume that the Parent population size is at least $\mu \geq n - 2k + 3$.

The next stage is to find the whole inner Pareto front. This process is essentially identical to the one of finding a corresponding segment of the Pareto front of the OMM problem. With foresight, we have proven in Theorem 4 in the OMM analysis a time bound for covering a subinterval of the Pareto front. Taking $\alpha = k$ and using exactly the same arguments, we obtain the following lemma.

Lemma 10. *Let T_2 be the number of iterations it takes to find the inner Pareto front once an inner Pareto optimum is in the population, that is,*

$$T_2 = \inf\{t \in \mathbb{Z}_{\geq 0} \mid F_I^* \subseteq f(P_t)\} - T_1.$$

Then $\mathrm{E}[T_2] = O(\frac{\mu n}{\lambda} + n + \frac{n^2}{\lambda} \log n)$.

It remains to find the two extremal points of the Pareto front. This analysis again builds heavily on our understanding of the population dynamics.

Lemma 11. *Let T_3 be the number of iterations the algorithm takes to find the two extremal Pareto optima once the inner part of the Pareto front is found, that is,*

$$T_3 = \inf\{t \in \mathbb{Z}_{\geq 0} \mid F^* \subseteq f(P_t)\} - T_2 - T_1.$$

Then, $\mathrm{E}[T_3] = O(1 + \frac{n^{k+1}}{\lambda} + \lceil \frac{\mu}{\lambda} \rceil \log \frac{\mu}{n-2k+3})$.

By adding up the runtime estimates for each stage (and a not very interesting technical argument needed to simplify the runtime expression), we obtain the main result of this subsection.

Runtime Analysis for LEADINGONESTRAILINGZEROS

We finally analyze in Theorem 12 how the SPEA2 optimizes the LOTZ benchmark, the third very popular benchmark in the field. We prove a runtime guarantee of an expected number of $O((\frac{\mu}{\lambda} + 1)n \log \frac{\mu}{n+1} + \frac{n^3}{\lambda} + n)$ iterations, or $O((\mu + \lambda)n \log \frac{\mu}{n+1} + n^3 + \lambda n)$ function evaluations, provided that the parent population size μ is at least the size of the Pareto front $n + 1$.

Theorem 12. *The expected runtime of the SPEA2 with parent population size $\mu \geq n + 1$ and offspring population size λ, optimizing the* LOTZ *benchmark is $O((\frac{\mu}{\lambda}+1)n \log \frac{\mu}{n+1} + \frac{n^3}{\lambda} + n)$ iterations, or $O((\mu + \lambda)n \log \frac{\mu}{n+1} + n^3 + \lambda n)$ function evaluations.*

We follow the usual and natural approach of analyzing separately the times to first reach the Pareto front and then to spread out there until the full Pareto front is covered (Laumanns, Thiele, and Zitzler 2004). We cannot use the technically easier appearing variant of (Doerr, Ivan, and Krejca 2025) that first regards the growth of the number of leading ones in the population and then reuses this analysis for the number of trailing zeros. The reason is that an objective value corresponding to a certain number of leading ones may be dominated by another one with same number of leading ones, resetting the multiplicative growth process that is the heart of all of our results. We therefore regard how the maximum sum of objective values in the population grows. This avoids the problem since now a domination implies an actual increase of this progress measure. The time to increase the sum of objectives is estimated in the following lemma.

Lemma 13. *For a set of individuals P, let $\nu(P) = \max\{f_1(x) + f_2(x) \mid x \in P\}$. Consider the SPEA2 with $\mu \geq n+1$ optimizing the* LOTZ *benchmark with $n \geq 3$. Assume that in some iteration $t_0 \in \mathbb{Z}_{\geq 0}$ we have $\nu(P_{t_0}) < n$.*

Let $T = \inf\{t \in \mathbb{Z}_{>t_0} \mid \nu(P_t) > \nu(P_{t_0})\}$. Then

$$\mathrm{E}[T - t_0] = O(\lceil \tfrac{\mu}{\lambda} \rceil \log \tfrac{\mu}{n+1} + \tfrac{n^2}{\lambda} + 1).$$

The second ingredient to our main proof is a statement on how long it takes to generate a neighboring Pareto-optimal objective value from an existing one. The arguments for this are essentially the same as in the previous lemma, even easier, since now we go from a fixed value v to a fixed value w and we know that v cannot be lost as it is Pareto-optimal.

Lemma 14. *Consider the SPEA2 with $\mu \geq n + 1$ optimizing the* LOTZ *benchmark with $n \geq 3$. Assume that in some iteration $t_0 \in \mathbb{Z}_{\geq 0}$, the population P_{t_0} contains an individual with Pareto-optimal objective value v. Let w be a neighboring Pareto-optimal solution value, that is, $w = (v_1 + 1, v_2 - 1)$ or $w = (v_1 - 1, v_2 + 1)$ provided this lies in $[0..n]^2$.*

Let $T = \inf\{t \in \mathbb{Z}_{\geq t_0} \mid w \in f(P_t)\}$. Then

$$\mathrm{E}[T - t_0] = O(\lceil \tfrac{\mu}{\lambda} \rceil \log \tfrac{\mu}{n+1} + \tfrac{n^2}{\lambda} + 1).$$

Using the two lemmas above, we obtain Theorem 12 simply by adding up suitable waiting times.

Conclusion

We conducted a rigorous runtime analysis of the SPEA2 on the three main benchmarks used in the runtime analysis community. Our results show that, in particular in comparison with the well-studied NSGA-II, the performance of the standard SPEA2 is less sensitive to the choice of its population size. Combined with our more general main arguments, these findings suggest that the SPEA2 can serve as a robust choice in practical applications where an optimal population size is unknown.

We note that the selection in the SPEA2 based on the σ-criterion is computationally more costly than the crowding distance employed by the NSGA-II. However, the crowding distance (with uniform tie-breaking) can fail already for easy problems with at least three objectives (Zheng and Doerr 2024a), and the alternative proposed by Doerr, Ivan, and Krejca (2025) adds complexity. A theoretical comparison of such added costs is an interesting step for future work.

Last, in the case of a population size smaller than the Pareto front size, our results do not apply. However, Alghouass et al. (2025) proved that the SPEA2 converges within reasonable time to a uniform spread across the Pareto front on the ONEMINMAX benchmark, whereas the NSGA-II does not. An interesting next step is to see whether our analysis of the population dynamics also results in improved convergence times for the SPEA2 in this setting.

Acknowledgments

This research benefited from the support of the FMJH Program PGMO. This work has profited from many scientific discussions at the Dagstuhl Seminars 23361 "Multiobjective Optimization on a Budget" and 24271 "Theory of Randomized Optimization Heuristics".

References

Alghouass, Y.; Doerr, B.; Krejca, M. S.; and Lagmah, M. 2025. Proven approximation guarantees in multi-objective optimization: SPEA2 beats NSGA-II. In *International Joint Conference on Artificial Intelligence, IJCAI 2025*, 8833–8841. ijcai.org.

Beume, N.; Naujoks, B.; and Emmerich, M. 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181: 1653–1669.

Bian, C.; Zhou, Y.; Li, M.; and Qian, C. 2023. Stochastic population update can provably be helpful in multi-objective evolutionary algorithms. In *International Joint Conference on Artificial Intelligence, IJCAI 2023*, 5513–5521. ijcai.org.

Coello, C. A. C.; Lamont, G. B.; and van Veldhuizen, D. A. 2007. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2nd edition.

Dang, D.; Opris, A.; and Sudholt, D. 2024. Crossover can guarantee exponential speed-ups in evolutionary multi-objective optimisation. *Artificial Intelligence*, 330: 104098.

Deb, K.; and Jain, H. 2014. An evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, part I: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18: 577–601.

Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6: 182–197.

Do, A. V.; Neumann, A.; Neumann, F.; and Sutton, A. M. 2023. Rigorous runtime analysis of MOEA/D for solving multi-objective minimum weight base problems. In *Advances in Neural Information Processing Systems, NeurIPS 2023*, 36434–36448.

Doerr, B. 2018. An elementary analysis of the probability that a binomial random variable exceeds its expectation. *Statistics and Probability Letters*, 139: 67–74.

Doerr, B.; Ivan, T.; and Krejca, M. S. 2025. Speeding up the NSGA-II with a simple tie-breaking rule. In *Conference on Artificial Intelligence, AAAI 2025*, 26964–26972. AAAI Press.

Doerr, B.; and Kötzing, T. 2021. Multiplicative up-drift. *Algorithmica*, 83: 3017–3058.

Doerr, B.; Krejca, M. S.; and Stanković, M. 2025. Improved Runtime Guarantees for the SPEA2 Multi-Objective Optimizer. *CoRR*, abs/2511.07150.

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

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

Doerr, B.; and Zheng, W. 2021. Theoretical analyses of multi-objective evolutionary algorithms on multi-modal objectives. In *Conference on Artificial Intelligence, AAAI 2021*, 12293–12301. AAAI Press.

Giel, O.; and Lehre, P. K. 2010. On the effect of populations in evolutionary multi-objective optimisation. *Evolutionary Computation*, 18: 335–356.

Laumanns, M.; Thiele, L.; and Zitzler, E. 2004. Running time analysis of multiobjective evolutionary algorithms on pseudo-Boolean functions. *IEEE Transactions on Evolutionary Computation*, 8: 170–182.

Li, Y.-L.; Zhou, Y.-R.; Zhan, Z.-H.; and Zhang, J. 2016. A primary theoretical study on decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20: 563–576.

Opris, A. 2025. A many objective problem where crossover is provably indispensable. In *Conference on Artificial Intelligence, AAAI 2025*, 27108–27116. AAAI Press.

Ren, S.; Bian, C.; Li, M.; and Qian, C. 2024a. A first running time analysis of the Strength Pareto Evolutionary Algorithm 2 (SPEA2). In *Parallel Problem Solving from Nature, PPSN 2024, Part III*, 295–312. Springer.

Ren, S.; Qiu, Z.; Bian, C.; Li, M.; and Qian, C. 2024b. Maintaining diversity provably helps in evolutionary multimodal optimization. In *International Joint Conference on Artificial Intelligence, IJCAI 2024*, 7012–7020. ijcai.org.

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

Wietheger, S.; and Doerr, B. 2024. Near-tight runtime guarantees for many-objective evolutionary algorithms. In *Parallel Problem Solving from Nature, PPSN 2024, Part IV*, 153–168. Springer.

Zheng, W.; and Doerr, B. 2023. Mathematical runtime analysis for the non-dominated sorting genetic algorithm II (NSGA-II). *Artificial Intelligence*, 325: 104016.

Zheng, W.; and Doerr, B. 2024a. Runtime analysis for the NSGA-II: proving, quantifying, and explaining the inefficiency for many objectives. *IEEE Transactions on Evolutionary Computation*, 28: 1442–1454.

Zheng, W.; and Doerr, B. 2024b. Runtime analysis of the SMS-EMOA for many-objective optimization. In *Conference on Artificial Intelligence, AAAI 2024*, 20874–20882. AAAI Press.

Zheng, W.; Liu, Y.; and Doerr, B. 2022. A first mathematical runtime analysis of the non-dominated sorting genetic algorithm II (NSGA-II). In *Conference on Artificial Intelligence, AAAI 2022*, 10408–10416. AAAI Press.

Zhou, A.; Qu, B.-Y.; Li, H.; Zhao, S.-Z.; Suganthan, P. N.; and Zhang, Q. 2011. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1: 32–49.

Zitzler, E.; Laumanns, M.; and Thiele, L. 2001. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK report*, 103.