Expected Runtimes of a Simple Multi-objective Evolutionary Algorithm

Oliver Giel[⋆]
FB Informatik, LS 2, Univ. Dortmund
44221 Dortmund, Germany
oliver.giel@uni-dortmund.de

Abstract- The expected runtime of a simple multi-objective evolutionary algorithm for the Boolean decision space is analyzed. The algorithm uses independent bit flips as mutation operator and, therefore, searches globally. It is proved that the expected runtime is $O(n^n)$ for all objective functions $\{0,1\}^n \to \mathbb{R}^m$. This worst-case bound is tight and matches the worst-case bounds for fundamental evolutionary algorithms working in the scenario of single-objective optimization. For the bicriteria problem LOTZ (Leading Ones Trailing Zeroes), it is shown that the expected runtime is $O(n^3)$. Moreover, the runtime is $O(n^3)$ with an overwhelming probability. Finally, the function $x \mapsto (x^2, (x-2)^2)$ that serves as a test function in the continuous decision space is adapted to the Boolean decision space, and bounds on the runtime are derived.

1 Introduction

Randomized search heuristics are applied to optimization problems in situations where problem-specific algorithms are not available. The lack of such algorithms can have various reasons. Problem-specific algorithms might be unknown for the considered problem, there might be not enough time and not enough experts to devise a problem-specific algorithm, or there might be only little knowledge about the structure of the problem. General search heuristics that do not employ problem-specific knowledge are of particular interest in theoretical investigations. In applications, these heuristics are often combined with problem-specific modules. Evolutionary algorithms (EAs) are such randomized search heuristics. They are not only applied to single-objective optimization problems but also to multi-objective optimization problems. Practical knowledge on the design of multi-objective EAs has increased considerably in recent years but theoretical works are rare. A common approach to learn how EAs work is to analyze basic EAs. In this work, we analyze the expected runtime of a very simple but fundamental multi-objective EA.

Theoretical analyses of the runtime of basic EAs in the scenario of single-objective optimization have been carried out in recent years. Most results giving time bounds consider discrete search spaces (e.g., Droste, Jansen, and Wegener (1998), Garnier, Kallel, and Schoenauer (1999), Droste, Jansen, and Wegener (2002), Wegener and Witt (2003)). Rigorous proofs on the runtime in a continuous search space have only been obtained recently (Jägersküpper (2003)). For an overview, refer to Wegener (2001) and Beyer, Schwefel, and Wegener (2002).

Works on the analysis of multi-objective EAs have mostly focused on the limit behavior (convergence), i.e., the question under what conditions an algorithm can find the set of optimal solutions when time goes to infinity (Rudolph (1998a,b, 2001), Rudolph and Agapie (2000)). It is not possible to derive sharp bounds on the (expected) runtime without taking into account some properties of the function (the problem) to be optimized. Scharnow, Tinnefeld, and Wegener (2002) have analyzed the expected runtime of a variant of the (1+1) EA on a multi-objective formulation of the single-source shortest-path problem. However, the objectives of the problem are non-conflicting. Laumanns, Thiele, Zitzler, Welzl, and Deb (2002) have been the first to analyze the (expected) runtime of two local search algorithms (SEMO and FEMO) for a problem with two conflicting objectives (LOTZ). In this work, we consider a closely related algorithm that searches globally by the use of a mutation operator that flips each bit independently.

The next section describes the scenario of multi-objective optimization in the framework of a partially ordered objective space and defines the goal of algorithms working in this scenario. Section 3 introduces the algorithm studied in subsequent sections and derives tight bounds on the expected runtime in the worst case. Sections 4 and 5 consider the (expected) runtime of the algorithm for some bicriteria example problems, including the LOTZ function.

2 Scenario and Basic Definitions

In the scenario of multi-objective optimization, m incommensurable and often conflicting objectives of a solution to some problem have to be optimized at the same time. The objective space F can be thought of as a set of real-valued vectors such that each of m components of a vector represents an objective of a solution. We assume all objectives to be maximized. Obviously, an objective vector x is not better than another vector y if each component of x is not larger than the corresponding component of y. However, one cannot tell which of two distinct vectors is better in general. There is no natural total order on the objective space if the objectives are incommensurable. In this scenario, the aim of optimization is to find solutions such that an improvement regarding one objective can only be achieved at the expense of another objective. We follow Rudolph (1998a, 2001) for basic definitions.

[⋆] Supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

Definition 1 (preorder, partial order). *Let F be a set and $\preceq$ a binary relation in F. The relation $\preceq$ is called a* preorder *if it is reflexive and transitive. The pair $(F, \preceq)$ is called a* partially ordered set (poset) *if $\preceq$ is an antisymmetric ($\forall x, y \in F : x \preceq y \wedge y \preceq x \Rightarrow x = y$) preorder. Distinct x and y are* incomparable, *denoted $x \parallel y$, if neither $x \preceq y$ nor $x \succeq y$. Otherwise, if $x \preceq y$ or $x \succeq y$, x and y are* comparable. *In particular, x is comparable to x.*

The relation in the set of real-valued vectors described above is a partial order.

Definition 2 (domination and maximal elements). *If $x \preceq y$, we say y* weakly dominates *x. We say y* dominates *x, denoted $x \prec y$, if $x \preceq y$ and $x \neq y$. An element $x^* \in F$ is called* maximal element *of the preordered set $(F, \preceq)$ if there is no $x \in F$ such that $x^* \prec x$. $M(F, \preceq)$ is the set of all maximal elements in $(F, \preceq)$.*

If F is a finite set, the set $M(F, \preceq)$ is finite and complete. $M(F, \preceq)$ is said to be *complete* if for each $x \in F$ there exists an $x^* \in M(F, \preceq)$ such that $x \preceq x^*$.

In the framework of multi-objective optimization without constraints, we have the *decision space X* (the set of all possible solutions), the partially ordered *objective space F* (the poset of objective vectors), and an *objective function* $f : X \rightarrow F$. The aim of multi-objective optimization is *not* to compute the set of maximal elements in the objective space. We are rather interested in a set of best solutions in the decision space, the preimage of the maximal elements in the objective space. As the objective function is generally not a bijection, the preimage might be empty or considerably large. We must take care with regard to the definition of the aim in solving a multi-objective optimization problem.

Definition 3 ($\preceq_f$). *Let X be the decision space and let $(F, \preceq)$ be the partially ordered objective space. Let $f : X \rightarrow F$ be a mapping. Then f induces a preorder $\preceq_f$ on X by the following definition:*

$$
\begin{aligned}
x \prec_f y \ &:\Leftrightarrow \ f(x) \prec f(y), \\
x =_f y \ &:\Leftrightarrow \ f(x) = f(y), \\
x \preceq_f y \ &:\Leftrightarrow \ x \prec_f y \ \vee \ x =_f y.
\end{aligned}
$$

In general, the preorder $\preceq_f$ is not a partial order since $x \preceq_f y \wedge y \preceq_f x \not\Rightarrow x = y$.

We use the notion of Pareto optimality if $f = (f_1, \ldots, f_m)$ is a vector-valued objective function, i.e., if F is a subset of $\mathbb{R}^m$.

Definition 4 (Pareto front, Pareto set). *Let X be a finite decision space, let $F := f(X) = \{f(x) \mid x \in X\} \subseteq \mathbb{R}^m$ be the objective space, and let the partial order $\preceq$ in F be defined by*

$$
(y_1, \ldots, y_m) \preceq (z_1, \ldots, z_m) \ \Leftrightarrow \ \forall i : y_i \leq z_i . \qquad (1)
$$

The set of all maximal elements $F^* = M(F, \preceq)$ in the objective space is called Pareto front. *An element $x \in X$ in the decision space is* Pareto optimal *if $f(x)$ belongs to the Pareto front F^*. The set of all Pareto optimal elements $X^* = f^{-1}(F^*)$ is called* Pareto set.

Definition 4 provides a surjective mapping f and ensures that the objective space $(F, \preceq)$ is a finite poset with a finite (and complete) set of maximal elements, the Pareto front. In the following, we assume the scenario of Definition 4.

Roughly speaking, the goal of multi-objective optimization is to compute the Pareto set X^*. This goal can be too ambitious if the Pareto set is fairly large. However, if the Pareto set is large (e.g., exponential size) and the Pareto front is small (e.g., polynomial size) there are solutions $x^1, \ldots, x^k$ with the same objective value $f(x^1) = \cdots = f(x^k)$. In this case, a set of solutions should imply only one solution $x \in \{x^1, \ldots, x^k\}$. Provided that the Pareto front is not too large, a set $A \subseteq X^*$ representing each objective value in the Pareto front F^* at least once is a reasonable set of solutions.

Definition 5 (approximation set). *A set $A' \subseteq F$ is called an* approximation set *(for the Pareto front) if no element in A' is weakly dominated by any other element in A' with respect to $\preceq$, i.e., any two distinct elements are incomparable.*

In Definition 5, we can replace weak domination by domination: For distinct elements in a poset, weak domination is equivalent to domination.

Definition 6 (set of representatives). *A* set of representatives *for a set $A' \subseteq F$ is a set $A \subseteq f^{-1}(A')$ such that $f(A) = A'$ and $|A| = |A'|$.*

In this work, the goal of an algorithm is to compute a set of representatives for the Pareto front. Clearly, if f is not injective on the Pareto set, the Pareto set is not a set of representatives for the Pareto front. The computed set of solutions will be a subset of the Pareto set.

3 The Algorithm

The following evolutionary algorithm requires that the decision space is $X = \{0, 1\}^n$ and that there is a partial order relation $\preceq$ defined in the objective space $F = f(X)$. In particular, it applies to the scenario of multi-objective optimization in the Boolean decision space. The idea of the algorithm is that for each point of time t, the population A_t is a set of representatives for the approximation set $f(A_t)$. The approximation set $f(A_t)$ is meant to approach the Pareto front F^* as t increases.

Algorithm 1 (global SEMO).

```
choose $x \in \{0,1\}^n$ uniformly at random
determine $f(x)$
$A \leftarrow \{x\}$
loop
  select $x \in A$ uniformly at random
  create $x'$ by flipping each bit of $x$ indep. with prob. $1/n$
  determine $f(x')$
  if $\forall z \in A : x' \not\preceq_f z$
    $A \leftarrow \{z \in A \mid z \not\preceq_f x'\} \cup \{x'\}$
  end if
end loop
```

An implementation of the set A needs to store search points x together with their objective values $f(x)$. For the ease of notation, this aspect is not explicitly expressed in the description of Algorithm 1. Obviously, the initial population $A_1 = \{x_1\}$ is a set of representatives for the approximation set $\{f(x_1)\}$. The loop can be interpreted in the following way. At time t, the algorithm adds the offspring x'_t to the population A_t if there is no element in A_t that weakly dominates x'_t, i. e., each element in A_t is either dominated by x'_t or incomparable to x'_t. If x'_t is added to A_t, all elements in A_t dominated by x'_t are removed from A_t at the same time, i. e., afterwards all elements of A_t are incomparable with respect to $\preceq_f$. Hence, at each point of time t, A_t is a set of representatives for the set $f(A_t)$. The latter set is an approximation set since for $x, y \in A_t, x \parallel_f y \Rightarrow f(x) \parallel f(y)$.

In applications, Algorithm 1 needs a stopping criterion. In this work, we are interested in the first point of time that the aim of the optimization process is reached and define the runtime of Algorithm 1 in the following way.

Definition 7 (runtime). *Let A_t, $t \in \mathbb{N}$, denote the population after the $(t-1)$th iteration of the loop, i. e., after t objective function evaluations. The random number T_f is the minimum t such that $f(A_t) = F^*$. T_f is called the* runtime *of Algorithm 1 for f.*

The assumption is that objective function evaluations are expensive and dominate the costs of all other operations in the loop. Then it is reasonable to call T_f the runtime of Algorithm 1 for f. This measure is well accepted, particularly for evolutionary algorithms. It is also used in theoretical analyses of algorithms working in the black-box scenario (Droste, Jansen, Tinnefeld, and Wegener (2003)). From a practical point of view, the measure may not be fair if the population A becomes very large. If $\Omega(|X|)$ elements of $(X, \preceq_f)$ are incomparable, the population may grow to size $\Omega(2^n)$. That means, the algorithm is only applicable if there is a much better bound for all $|A_t|$.

Algorithm 1 is almost the same algorithm as the one studied in Laumanns, Thiele, Zitzler, Welzl, and Deb (2002) – there called SEMO (Simple Evolutionary Multi-objective Optimizer). The only difference is the mutation operator. Algorithm 1 flips each bit independently with probability $1/n$ whereas SEMO flips exactly one bit. That means SEMO searches locally in the manner of a hill climber. If there is a subset of non-Pareto optimal points $\widehat{X}$ in the search space such that all Hamming neighbors of these points outside $\widehat{X}$ are (weakly) dominated by the points in $\widehat{X}$, SEMO's population cannot escape from $\widehat{X}$ if the entire population is contained in $\widehat{X}$. The following example problem with two objectives shows that this can happen with an overwhelming probability. The first objective is the number of ones in a solution x if this number is even, otherwise it is 0. The second objective is the number of zeroes if this number is strictly less than $(1/4)n$, otherwise it is 0. Clearly, Pareto optimal solutions have at least $(3/4)n$ ones. By Chernoff bounds (e. g., Motwani and Raghavan (1995)), the probability of choosing an initial string x with $i < (2/3)n$ ones is $1 - e^{-\Omega(n)}$, i. e., exponentially close to 1. If this happens and i is odd, the objective value of x is $(0, 0)$. The Hamming neighbors have objective values $(i-1, 0)$ or $(i+1, 0)$ and dominate x. One of them is created first and replaces x in the population. Now, either by the initial step or by the first mutation, the algorithm is in the situation that the number of ones in the only individual x in the population is even and at most $(2/3)n$. All Hamming neighbors of x have objective values $(0, 0)$ and are dominated by x. Hence, the population gets stuck with an overwhelming probability before it reaches any point in the Pareto set. Therefore, SEMO has no finite expected runtime in the general case of an arbitrary f. In our example, restarts (multiple runs) do not help much since the probability of choosing a bad initial point is exponentially close to 1. The same applies to a variant of SEMO in Laumanns, Thiele, Zitzler, Welzl, and Deb (2002) called FEMO (Fair Evolutionary Multi-objective Optimizer). We conclude that local search strategies like SEMO and FEMO can only be applied if we have some intuition of the optimization problem that suggests that such strategies are not very likely to get trapped.

In contrast to local search strategies, Algorithm 1 searches globally, and its population will not get stuck in local optima forever. Local search strategies are typically easier to analyze than global search strategies. Nevertheless, the analysis of local search strategies can give insight into the problem at hand and is often a good starting point for the analysis of global search strategies. Our first step is to study the expected runtime of Algorithm 1 in the worst case, i. e., the expected runtime if the objective function is chosen by an adversary. It is easy to see that for $n = 1$, the runtime of Algorithm 1 is at most 2. In the remainder of this paper, we assume that the dimension n of the Boolean decision space is at least 2.

Theorem 1. *For any $f\colon \{0,1\}^n \to \mathbb{R}^m$, the expected runtime $E(T_f)$ is bounded above by $(1 + o(1))n^n$. There are functions f where $E(T_f) \geq n^n$.*

We need the following lemma in the proof of Theorem 1.

Lemma 1. *Given a set A of at least $n^{2\lceil \log n \rceil}$ points in $\{0,1\}^n$ and a point $x \in \{0,1\}^n \setminus A$. For more than half of the points in A, the Hamming distance to x is at most $n - \lceil \log n \rceil$.*

Proof. The number of points $y \in \{0,1\}^n$ with Hamming distance $H(y, x) = k$ is $\binom{n}{k}$. The number of points with a Hamming distance to x of at least $n - \lceil \log n \rceil + 1$ is

$$\sum_{n-\lceil \log n \rceil + 1 \leq k \leq n} \binom{n}{k} \leq \lceil \log n \rceil \binom{n}{\lceil \log n \rceil} \leq n^{\lceil \log n \rceil}$$

Consequently, A contains at least $n^{2\lceil \log n \rceil} - n^{\lceil \log n \rceil} > (1/2)n^{2\lceil \log n \rceil}$ points y with $H(x, y) \leq n - \lceil \log n \rceil$. □

Proof of Theorem 1. Whenever Algorithm 1 produces an offspring y such that $f(y)$ is in the Pareto front F^* and $f(y) \notin f(A)$, the algorithm adds y to A. As y is Pareto optimal, y will never be removed from A. Before A is a set of representatives for the Pareto front, such an individual y exists. At time t, let $Y_t \subseteq X^*$ be the set of Pareto optimal decision vectors whose corresponding objective values are not yet represented by any decision vector in A_t. Formally, we define the target set Y_t by $Y_t := X^* \setminus f^{-1}(f(A_t))$. The algorithm would accept each $y_j \in Y_t$ in the next mutation step. During a run of the algorithm, $Y_t \supset Y_{t+1}$ holds only if a new Pareto optimal point is added to A_t and otherwise $Y_t = Y_{t+1}$. We define A_0 to be the empty population in the initialization step (Step 0) and, therefore, $Y_0 = X^*$. Let $X^* = Y_0 \supseteq \cdots \supseteq Y_{T_f} = \emptyset$ be the random sequence of target sets produced by the algorithm. Note that, for $k = |F^*|$, there are exactly $k+1$ mutually distinct sets $X^* = Y_{i_k} \supset \cdots \supset Y_{i_0} = \emptyset$ in this sequence and $|Y_{i_j}| \ge j$. Let $E(T_{i_j})$ denote the expected number of steps spent for the set Y_{i_j}. Then the expected runtime is

$$E(T_f) = \sum_{k \ge j \ge 2} E(T_{i_j}) + E(T_{i_1}).$$

The probability that the initial step selects a Pareto optimal search point is $\frac{|Y_{i_k}|}{2^n} \ge \frac{n-1}{n^n}(|Y_{i_k}| - 1)$. For $t \ge 1$, let $x \in A_t$ denote the individual selected for mutation and let $Y_t = \{y_1, \ldots, y_{|Y_t|}\}$ be the target set at time t. There is at most one $y_j \in Y_t$ such that the Hamming distance $H(x, y_j) = n$, namely if $y_j = \overline{x}$. In all other cases, $H(x, y_j)$ is at most $n-1$. Hence, the probability that the algorithm creates an offspring in Y_t is lower bounded by

$$\begin{aligned} &\sum_{1 \le j \le |Y_t|} (1/n)^{H(x,y_j)}(1-1/n)^{n-H(x,y_j)} \\ \ge &\sum_{1 \le j \le |Y_t|-1} (1/n)^{n-1}(1-1/n) \;=\; \frac{n-1}{n^n}(|Y_t|-1), \end{aligned}$$

and for $|Y_{i_j}| \ge 2$ the expected value $E(T_{i_j})$ is at most $\frac{n^n}{(n-1)(|Y_{i_j}|-1)}$. Since $|Y_{i_j}| \ge j$, we have

$$E(T_f) \le \frac{n^n}{n-1} \sum_{k \ge j \ge 2} \frac{1}{j-1} + E(T_{i_1}). \tag{2}$$

Now we estimate the right-hand side of the last equation according to two cases.

The first case is $k \ge n^{2\lceil \log n \rceil} + 1$. We consider the steps when the target set is Y_{i_1}, i.e., the algorithm has discovered $k-1$ representatives for $k-1 = |F^*|-1$ points of the Pareto front before. Hence, $|A| = k-1 \ge n^{2\lceil \log n \rceil}$, and there is only one point x' in the Pareto front that is not in $f(A)$. Let $x \in f^{-1}(x') = Y_{i_1}$. By Lemma 1, the probability that the algorithm selects an individual in A such that the Hamming distance to x is at most $n - \lceil \log n \rceil$ is at least $1/2$. Thus, $E(T_{i_1})$ is bounded above by

$$\left((1/2)(1/n)^{n-\lceil \log n \rceil}(1-1/n)^{\lceil \log n \rceil}\right)^{-1} \le 2en^{n-\lceil \log n \rceil}.$$

Using $k = |F^*| \le |X^*| \le 2^n$, we can upper bound (2) by

$$\frac{n^n}{n-1} \sum_{j=2}^{2^n} \frac{1}{j-1} + 2en^{n-\lceil \log n \rceil} \le n^n \left(\frac{H_{2^n-1}}{n-1} + \frac{2e}{n^{\lceil \log n \rceil}} \right).$$

The last expression is strictly smaller than n^n for n large enough since the harmonic number H_{2^n-1} is bounded by $\ln(2^n - 1) + 1 \le 0.7n + 1$.

The second case is $k \le n^{2\lceil \log n \rceil}$. When the target set is Y_{i_1}, the probability that the next mutation step creates a point in this set is at least $1/n^n$. In the initial step, the probability is at least $k/2^n \ge 1/n^n$. Hence, $E(T_{i_1}) \le n^n$, and (2) is bounded by

$$\frac{n^n}{n-1} \sum_{j=2}^{k} \frac{1}{j-1} + n^n = n^n \left(\frac{H_{k-1}}{n-1} + 1 \right) = (1 + o(1))n^n,$$

using $H_{k-1} \le \ln(k-1) + 1 \le 0.7 \log n^{2\lceil \log n \rceil} + 1 \le 1.4\lceil \log n \rceil^2 + 1$.

For the lower bound, we consider the function

$$f(x) = \left(\prod_{1 \le i \le n} x_i, \prod_{1 \le i \le n} (1 - x_i) \right).$$

Obviously, the objective space is $F = \{(0,0), (1,0), (0,1)\}$ and only $(0,0)$ is not maximal. Let x be the Pareto optimal decision vector found first by the algorithm, i.e., either $x = 0^n$ or $x = 1^n$. Since x dominates all decision vectors found before, the population now is $A = \{x\}$. The Hamming distance to the second Pareto optimum $\overline{x}$ is n. Hence, the expected waiting time is n^n. □

Theorem 1 states a $\Theta(n^n)$ bound in the worst case. Note that the upper bound is independent of the number of objectives m and that the lower bound is obtained from a bicriteria problem. The scenario of multi-objective optimization includes the scenario of single-objective optimization. The (1+1) EA is perhaps the most fundamental evolutionary algorithm for single-objective optimization in the Boolean decision space $\{0,1\}^n$. Interestingly, it has the same expected runtime $\Theta(n^n)$ in the worst case (Droste, Jansen, and Wegener (2002)). If applied to a monocriteria problem, Algorithm 1 behaves almost like the (1+1) EA. One can also obtain $\Omega(n^n)$ bounds for Algorithm 1 from some monocriteria problems that have been analyzed for the (1+1) EA, e.g., the problem DISTANCE considered in Droste, Jansen, and Wegener (2002).

4 LOTZ – Leading Ones Trailing Zeroes

The LOTZ function has been studied in Laumanns, Thiele, Zitzler, Welzl, and Deb (2002) for the algorithms SEMO and a variant of SEMO called FEMO. Flipping exactly 1 bit in each step simplifies the analysis of these algorithms for this function. The effect is that the population size is bounded by 1 until the first point in the Pareto set is discovered, and then the algorithms explore the Pareto set without accepting solutions that are not Pareto optimal. Both properties do not carry over to Algorithm 1. The selection mechanism of FEMO has been adapted to the LOTZ function, and in fact FEMO performs better on LOTZ. The expected runtimes for SEMO and FEMO are $\Theta(n^3)$ and $\Theta(n^2 \log n)$, respectively. In this section, we show that using independent bit flips with SEMO (i.e., Algorithm 1) does not increase the runtime substantially. Moreover, the runtime is $O(n^3)$ with a probability exponentially close to 1. It is not known whether independent bit flips increase the runtime of FEMO for LOTZ.

Definition 8. *The functions* $\mathrm{LO}, \mathrm{TZ}\colon \{0,1\}^n \to \mathbb{N}$ *and* $\mathrm{LOTZ}\colon \{0,1\}^n \to \mathbb{N}^2$ *are defined by*

$$\begin{aligned}\mathrm{LO}(x) &:= \sum_{i=1}^{n}\prod_{j=1}^{i} x_j,\\ \mathrm{TZ}(x) &:= \sum_{i=1}^{n}\prod_{j=i}^{n}(1-x_j),\\ \mathrm{LOTZ}(x) &:= \big(\mathrm{LO}(x), \mathrm{TZ}(x)\big).\end{aligned}$$

$\mathrm{LO}(x)$ is the number of leading ones in x and $\mathrm{TZ}(x)$ the number of trailing zeroes. We define the relation $\preceq$ in $\mathbb{N}_0^2$ according to (1) and consider the partially ordered objective space $(\mathbb{N}_0^2 \cap \mathrm{LOTZ}(\{0,1\}^n), \preceq)$ and the preordered decision space $(\{0,1\}^n, \preceq_{\mathrm{LOTZ}})$. In the remainder of this section, we omit the subscript “LOTZ” in our notation.

Proposition 1. *The Pareto front* F^* *is the set* $\{(i, n-i) \mid 0 \le i \le n\}$*, and the Pareto set* X^* *is the set of all strings* $1^i 0^{n-i}$*,* $0 \le i \le n$*. The Pareto set* X^* *is the only set of representatives for the Pareto front* F^**.*

Proof. The set $\{(i,j) \mid 0 \le i+j \le n,\ i+j \ne n-1\}$ is the objective space, and only the elements $(i, n-i)$, $0 \le i \le n$, are not dominated by any other element. Obviously, $\mathrm{LOTZ}^{-1}(i, n-i)$ is the singleton set $\{1^i 0^{n-i}\}$. □

Proposition 2. *Let* A *be a set of representatives for an approximation set* A'*. The cardinality of* A *is at most* $n+1$*. If* $A \ne X^*$*, the cardinality of* A *is at most* n*.*

Proof. As $|A| = |A'|$, it suffices to show $|A'| \le n+1$. The characteristic function of the objective space $F \subseteq \{0,\dots,n\}^2$ can be viewed as a triangular matrix with 1-entries at (i,j), $0 \le i+j \le n$ and $i+j \ne n-1$. The row index i gives the number of leading ones, the column index j the number of trailing zeroes. Since A' is an approximation set, i.e., no element in A' is dominated by any other element in A', there is at most one element from each of the $n+1$ rows in A'. The same applies to columns. This shows that $|A'| \le n+1$. Assume $|A'| = n+1$. Then A' chooses exactly one element in each row and each column. Hence, the characteristic function of A' can be viewed as a permutation matrix. As $A' \subseteq F$, the 1-entries in the permutation matrix are also 1-entries in the triangular matrix representing F. It is easy to see that there is only one choice for A', namely all elements placed on the diagonal $(i, n-i)$, $0 \le i \le n$. Hence, $A' = F^*$. □

Theorem 2. *The expected runtime of Algorithm 1 for* LOTZ *is* $O(n^3)$*. The runtime is* $O(n^3)$ *with a probability* $1 - e^{-\Omega(n)}$*.*

Proof. As the Pareto set X^* is the unique set of representatives for F^* (Proposition 1), the population becomes static if $A = X^*$. It can change at any time before this event happens. We discern two epochs in a typical run of the algorithm. The first epoch starts after the initialization and is finished by the step producing the first individual $x \in X^*$. The following epoch lasts until $A = X^*$.

First we show that a phase of $s := \lceil en^3 \rceil$ steps finishes the first epoch with a probability $1 - e^{-\Omega(n)}$. We consider the initial individual x^0 and the (random) sequence of individuals $x^1, x^2, x^3, \dots$ in the first epoch such that x^{i+1} causes x^i to leave the population (because $x^{i+1} \succ x^i$). When the dominating individual x^{i+1} is created, either the number of leading ones compared to x^i is increased and the number of trailing zeros compared to x_i is not decreased or vice versa. That implies that there are at most n individuals in the above sequence that starts with x^1. If an offspring dominates its parent then it will be accepted and replace the parent individual. We estimate the probability to create x^{i+1} in the next step by the probability that x^i is chosen for mutation and the algorithm flips either only the leftmost 0 or only the rightmost 1. We call this event a success. Using Proposition 2, the probability of a success is at least $(1/n) \cdot 2 \cdot (1/n) \cdot (1 - 1/n)^{n-1} \ge 2/(en^2)$. Within the first phase, the expected number of successes is at least $2n$. By Chernoff bounds, the probability of less than n successes is $e^{-\Omega(n)}$. The first phase of s steps finishes the first epoch with a probability exponentially close to 1. To obtain an upper bound on the expected number of steps, we observe that our estimations also hold if we start a new phase with a population of up to n non-optimal solutions. The expected number of phases is upper bounded by 2. This implies that the expected number of steps in the first epoch is $O(s) = O(n^3)$.

Next we show that, starting with at least one Pareto optimal element in A, after a phase of $s' = \lceil 2en^3 \rceil$ steps, $X^* = A$ with a probability $1 - e^{-\Omega(n)}$. The Pareto set can be viewed as a path from 0^n to 1^n that visits all strings $1^i 0^{n-i}$, $0 \le i \le n$. Obviously, each individual on the path has at least one Hamming neighbor on the path. As long as $A \ne X^*$, there exists at least one $x \in A$ with a Hamming neighbor $x' \in X^* \setminus A$ and by Proposition 2, $|A| \le n$ holds. The probability of creating x' in the next step is at least $(1/n) \cdot (1/n) \cdot (1 - 1/n)^{n-1} \ge 1/(en^2)$. Within s' steps of a phase, the expected number of such successes is at least $2n$. Using Chernoff bounds again, the probability of less than n successes is $e^{-\Omega(n)}$. Analogously to the first epoch, the expected number of steps is $O(n^3)$, too.

Combining the results for both epochs yields the bounds in the theorem. □

5 A Test Function

The functions considered in this section are inspired by the well-known function $x \mapsto (x^2, (x-2)^2)$. The latter often serves as a test function for algorithms that work in the continuous decision space $\mathbb{R}$ (e.g., Srinivas and Deb (1994)). We adapt this function to the Boolean decision space in two different ways. The first variant uses a kind of unary encoding of integer numbers, the second one the standard binary encoding.

Definition 9. *For* $x = x_{n-1}, \dots, x_0 \in \{0,1\}^n$*, let* $\|x\| = \sum_{0 \le i \le n-1} x_i$ *denote the number of ones in* x *and* $\mathrm{BV}(x) = \sum_{0 \le i \le n-1} x_i 2^i$ *the binary value of* x*. The functions* $f_{a,b}$*,*

$0 \le a < b \le n$, and $g_{a,b}$, $0 \le a < b \le 2^n - 1$, are defined by

$$f_{a,b}(x) = \big((\|x\| - a)^2, (\|x\| - b)^2\big),$$
$$g_{a,b}(x) = \big((\mathrm{BV}(x) - a)^2, (\mathrm{BV}(x) - b)^2\big).$$

For both functions, the goal is to minimize the two objectives. We adapt the basic definitions to the case of minimization. In particular, we redefine (1) by $y \preceq z :\Leftrightarrow \forall i\colon\ y_i \ge z_i$.

Proposition 3. *The Pareto set and Pareto front of $f_{a,b}$ are $X^* = \{x \mid a \le \|x\| \le b\}$ and $F^* = \{(i^2, (b-a-i)^2) \mid 0 \le i \le b-a\}$, respectively.*

Proof. Any point x with $\|x\| < a$ ($\|x\| > b$) is not Pareto optimal since the value of both objectives decreases as $\|x\|$ increases (decreases) by 1. Consider a point z, $a \le \|z\| \le b$. We show that z is not dominated by any point w, i.e., z is Pareto optimal. If $\|z\| = \|w\|$ then $z =_{f_{a,b}} w$. If $\|z\| < \|w\|$ then $(\|z\| - a)^2 < (\|w\| - a)^2$ holds and implies $z \not\preceq_{f_{a,b}} w$. If $\|z\| > \|w\|$ then $(\|z\| - b)^2 < (\|w\| - b)^2$ holds and implies $z \not\preceq_{f_{a,b}} w$. For all z with $\|z\| = a + i$, the corresponding objective vector is $(((a+i) - a)^2, ((a+i) - b)^2)$. ☐

Proposition 4. *The Pareto set and Pareto front of $g_{a,b}$ are $X^* = \{x \mid a \le \mathrm{BV}(x) \le b\}$ resp. $F^* = \{(i^2, (b-a-i)^2) \mid 0 \le i \le b-a\}$, and $|X^*| = |F^*|$.*

Proof. Can be carried out analogously to the proof of Proposition 3. ☐

Theorem 3. *The expected runtime of Algorithm 1 for $f_{a,b}$ is*

$$O\big(n \log n + n(b-a)\log(b-a)\big).$$

Note that $b - a + 1$ is the cardinality of the Pareto front. If a and b are constants (as in $x \mapsto (x^2, (x-2)^2)$), the expected runtime is $O(n \log n)$.

Proof. We partition the process into two epochs. The first epoch is the time before the first search point in the Pareto set is produced and the second epoch is the remaining time until the image of the population is exactly the Pareto front.

For the first epoch, note that the population size $|A|$ is bounded by 2. At any time, there is at most one search point x^{low} such that $\|x^{\text{low}}\| < a$ because any other point with this property would either dominate x^{low} or be dominated by x^{low}. For the same reason, there is at most one search point x^{high} such that $\|x^{\text{high}}\| > b$. Let $x \in A$ in the first epoch. From a local point of view, the aim for x is to increase (decrease) the number of ones if $\|x\| < a$ ($\|x\| > b$) . If we consider only x, the scenario is similar to the situation where the (1+1) EA optimizes the function OneMax (ZeroMax). The expected runtime of the (1+1) EA for OneMax is known to be $\Theta(n \log n)$ (Droste, Jansen, and Wegener (2002)). We make the upper bound for Algorithm 1 explicit here. At any time, let x be the individual in the population such that $d = |\|x\| - a|$ takes the smaller value; ties broken arbitrarily. The d-value is non-increasing in the first epoch since d^2 is the first objective of individual x, and x can only be dominated by a new individual with a d-value that is not larger. Notice that we can always specify d bits of x such that flipping these bits would decrease the d-value to 0. The probability that the d-value decreases in the next step is lower bounded by the probability that the next step chooses x for mutation and flips solely one bit out of d specified bits in x. The latter probability is at least $(1/2)d(1/n)(1-1/n)^{n-1} \ge d/(2en)$. The expected time until the first Pareto optimal search point is produced is at most

$$\sum_{n \ge d \ge 1} \frac{2en}{d} = 2enH_n = O(n \log n),$$

where H_n denotes the nth harmonic number.

We argue that for each point of time $t < T_{f_{a,b}}$ in the second epoch, the size of the population is at most $b - a$. If there are only Pareto optimal search points in the population, this property follows from Proposition 3 as $A \neq X^*$ in the second epoch. Now consider the case that there are non-Pareto optimal points in A. We have already seen that there are at most two individuals $x^{\text{low}}, x^{\text{high}} \in \{0,1\}^n - X^*$ in the population, where $\|x^{\text{low}}\| < a$ and $\|x^{\text{high}}\| > b$. If only one of them exists, say x^{low} (implying $a \ge 1$), then we have to show that there are strictly less than $b - a$ points of the Pareto set in the population. As the minimum value of the first objective of x^{low} is 1, we can exclude at least all points in X^* whose first objective takes a value of 0 or 1, i.e., all points $x \in X^*$ with $\|x\| \in \{a, a+1\}$. The remaining $b-a-1$ points in the Pareto front are represented by at most that many individuals. If only x^{high} exists (implying $b < n$), we can exclude all points $x \in X^*$ with $\|x\| \in \{b-1, b\}$ using analogous arguments. If both x^{low} and x^{high} exist, we can exclude all points in X^* with $\|x\| \in \{a, a+1, b-1, b\}$. In the last case, there are at most $b - a - 3$ Pareto optimal points in the population plus x^{low} and x^{high}.

Now we estimate the probability p_i, $a \le i \le b$, that a search point with i ones is created in the next step, given that there is already a search point x in the population with $i-1$ or $i+1$ ones. The algorithm selects x for mutation with a probability of at least $1/(b-a)$. The probability that the mutation step creates a string with exactly i ones from a string with $i-1$ ones is at least $(n-i+1)(1/n)(1-1/n)^{n-1}$; the probability that the mutation step creates such a string from a string with $i+1$ ones is at least $(i+1)(1/n)(1-1/n)^{n-1}$. We only underestimate the probability p_i if we use the bounds

$$p_i \ge \begin{cases} \frac{n-i+1}{b-a}\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \ge \frac{n-i+1}{(b-a)ne} & \text{if } i > n/2, \\ \frac{i+1}{b-a}\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \ge \frac{i+1}{(b-a)ne} & \text{if } i \le n/2. \end{cases}$$

Let T_i denote the waiting time until a string with i ones is created given that a string with $i-1$ or $i+1$ ones has been created before. Then $E(T_i)$ is at most $1/p_i$. Let j be the number of ones of the first point in the Pareto set created by the algorithm. The expected duration of the second epoch is at most

$$E(T_{j-1}) + \cdots + E(T_a) + E(T_{j+1}) + \cdots + E(T_b).$$

Our bounds for $E(T_0)$ and $E(T_n)$ are the largest, namely $(b-a)ne/1$. For $E(T_1)$ and $E(T_{n-1})$, they are $(b-a)ne/2$

and so on. Hence, the last sum is upper bounded by the sum of the $b-a$ largest bounds. The latter is at most

$$(b-a)ne \cdot 2\left(\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{\lceil (b-a)/2\rceil}\right)$$
$$= 2en(b-a)H_{\lceil (b-a)/2\rceil} \;=\; O\big(n(b-a)\log(b-a)+n\big).$$

□

The function becomes much harder for Algorithm 1 if we switch to the standard bit representation of integer numbers. The reason is that there can be large Hamming cliffs in the Pareto set. We show that, for each $d \in \{2,\ldots,n\}$, one can choose a and b such that the expected runtime is $\Theta(n^d)$.

Theorem 4. *Let* $a = 2^k - 1$, $b = 2^k$, *and* $1 \le k \le n-1$. *The expected runtime of Algorithm 1 for* $g_{a,b}$ *is* $\Theta(n^{k+1})$. *Moreover, for* $0 < c < 1$, *the runtime is at least* $n^{c(k+1)}$ *with a probability of at least* $1 - \frac{1}{n^{(1-c)(k+1)}}$ *and at most* n^{k+2} *with a probability* $1 - e^{-\Omega(n)}$.

Proof. According to Proposition 4, the Pareto set is $X^* = \{\mathrm{BV}^{-1}(a), \mathrm{BV}^{-1}(b)\} = \{0^{n-k}1^k, 0^{n-k-1}10^k\}$, and the Pareto front is $F^* = \{(0,1),(1,0)\}$. ($\mathrm{BV}^{-1}(i)$ denotes the bit representation of a non-negative integer i.) We partition the run of the algorithm into two consecutive epochs. The first epoch lasts until the first Pareto optimal search point $\mathrm{BV}^{-1}(a)$ or $\mathrm{BV}^{-1}(b)$ is created. For the first epoch, we prove only a weak upper bound because the second epoch dominates the runtime; however, we must take care that our bound holds with a probability $1 - e^{-\Omega(n)}$.

Note that the population size $|A|$ is bounded by 2 in the first epoch. There is at most one search point x^{low} such that $\mathrm{BV}(x^{\text{low}}) < a$ because any other point with this property would either dominate x^{low} or be dominated by x^{low}. For the same reason, there is at most one search point x^{high} such that $\mathrm{BV}(x^{\text{high}}) > b$. We subdivide the first epoch into two subepochs such that the first subepoch lasts until an individual x with $\mathrm{BV}(x) < 2^{k+1}$ is created. Clearly, the population is $A = \{x^{\text{high}}\}$ in the first subepoch. If solely the leftmost one of x^{high} flips, $\mathrm{BV}(x^{\text{high}})$ is at least halved. We call this event a success in the first subepoch. The probability that a step is a success is $(1/n)(1-1/n)^{n-1} \ge 1/(ne)$. As the initial value of $\mathrm{BV}(x^{\text{high}})$ is at most $2^n - 1$, a number of $n-(k+1)$ successes are sufficient in the first subepoch. In the second subepoch, the binary value of each individual in A is less than 2^{k+1}, i.e., all prefix bits corresponding to the weights $2^{n-1},\ldots,2^{k+1}$ are 0-bits. For $x \in A$, let $d(x) := \min\{|\,\mathrm{BV}(x)-a|, |\,\mathrm{BV}(x)-b|\}$, and let $d(A) = \min\{d(x), x \in A\}$, i.e., $d(A)$ is the smallest distance from a point in A to a point in the Pareto set in terms of binary values. At any time, let $x \in A$ be the point with the smaller $d(x)$-value; ties broken arbitrarily. Note that if x is removed from A, a new individual with a d-value that is not larger enters the population at the same time. Consequently, the $d(A)$-value only decreases with time. If $\mathrm{BV}(x) > b$ then $x_k = 1$ and $d(A) = \mathrm{BV}(x) - b = \sum_{0\le i\le k-1} x_i 2^i$. Flipping solely the leftmost 1-bit in the suffix $x_{k-1},\ldots,x_0$ reduces the $d(A)$-value at least by a factor of $1/2$. If $\mathrm{BV}(x) < a$ then $x_k = 0$ and $d(A) = a - \mathrm{BV}(x) = \sum_{0\le i\le k-1}(1-x_i)2^i$. Flipping solely the leftmost 0-bit in the suffix $x_{k-1},\ldots,x_0$ reduces the $d(A)$-value at least by a factor of $1/2$. The algorithm selects x for mutation with a probability of at least $1/2$. Hence, the next step decreases the $d(A)$-value at least by a factor of $1/2$ with a probability of at least $(1/2)(1/n)(1-1/n)^{n-1} \ge 1/(2en)$. We call this event a success in the second subepoch. A number of k successes are sufficient for the second subepoch, and less than n successes are sufficient for the first epoch. In a sequence of $12n^2$ steps, the expected number of successes is at least $2n$ and, by Chernoff bounds, the probability of less than n successes is $e^{-\Omega(n)}$.

By the time that the second epoch starts, the algorithm has found either $\mathrm{BV}^{-1}(a)$ or $\mathrm{BV}^{-1}(b)$ first. The point $\mathrm{BV}^{-1}(a)$ ($\mathrm{BV}^{-1}(b)$) dominates all other points in the decision space except $\mathrm{BV}^{-1}(b)$ ($\mathrm{BV}^{-1}(a)$). Therefore, the population is $\{\mathrm{BV}^{-1}(a)\}$ or $\{\mathrm{BV}^{-1}(b)\}$, and no offspring except $\mathrm{BV}^{-1}(b)$ resp. $\mathrm{BV}^{-1}(a)$ will be accepted. Only a mutation step flipping solely the $k+1$ rightmost bits corresponding to the weights $2^k,\ldots,2^0$ will be accepted. The corresponding probability is $(1/n)^{k+1}(1-1/n)^{n-(k+1)}$. It is upper and lower bounded by $1/n^{k+1}$ and $1/(en^{k+1})$, respectively. Thus, the expected waiting time for this event is upper and lower bounded by the expectations of random variables following the geometric distribution with parameter $1/(en^{k+1})$ and $1/(n^{k+1})$, respectively. Hence, the expected runtime (for both epochs) is $\Theta(n^{k+1})$.

The probability that a number of steps in the second epoch succeeds in producing the second Pareto optimal point is upper bounded by the sum of the success probabilities in each step. Hence, the probability that the first $n^{c(k+1)}$ steps in the second epoch are not successful is lower bounded by

$$1 - \frac{1}{n^{(k+1)}} n^{c(k+1)} = 1 - \frac{1}{n^{(1-c)(k+1)}}.$$

Remember that the first epoch is finished after $12n^{k+1}$ steps with an overwhelming probability $1 - e^{-\Omega(n)}$. The first $n^{k+2} - 12n^{k+1}$ steps in the second epoch succeed in finding the second Pareto optimum with a probability of at least

$$1 - \left(1 - \frac{1}{en^{k+1}}\right)^{n^{k+2}-12n^{k+1}} \ge 1 - e^{-\Omega(n)}.$$

□

Although one of the two Pareto optima is found quickly by the algorithm (almost surely in time $O(n^2)$), a large Hamming distance to the second Pareto optimum ensures a large (expected) runtime. For $k = \Theta(n)$, the runtime is $n^{\Theta(n)}$ with a probability exponentially close to 1. Apparently, Algorithm 1 would not always find the same Pareto optimum first. Multiple runs could help to detect the entire Pareto set if each instance of the algorithm is halted after $12n^2$ steps and non-dominated solutions in the union of the final populations are computed.

6 Conclusion

The expected runtime of simple multi-objective EAs that search globally can be analyzed. In the worst case, the presented algorithm has an expected runtime that matches the expected worst-case runtime of simple EAs working in the scenario of single-objective optimization. Explicit time bounds for simple objective functions can be derived, e. g., for the LOTZ function. For each $d \in \{2, \ldots, n\}$, we have exhibited a function such that the expected runtime of the algorithm is $\Theta(n^d)$.

Acknowledgments

I am grateful to Marco Laumanns for discussions on multi-objective EAs and their analysis. I thank Carsten Witt for helpful comments and Ingo Wegener, who gave a hint that improved the upper bound in Theorem 1 from $3n^n$ to $(1 + o(1))n^n$.

Bibliography

Beyer, H.-G., Schwefel, H.-P., and Wegener, I. (2002). How to analyse evolutionary algorithms. *Theoretical Computer Science* 287, 101–130.

Droste, S., Jansen, T., Tinnefeld, K., and Wegener, I. (2003). A new framework for the valuation of algorithms for black-box optimization. *Proc. of the 7th Foundations of Genetic Algorithms Workshop (FOGA 7)*, 253–270.

Droste, S., Jansen, T., and Wegener, I. (1998). On the optimization of unimodal functions with the (1+1) evolutionary algorithm. *Proc. of the 5th Conf. on Parallel Problem Solving from Nature (PPSN V),* LNCS 1498, 13–22.

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

Garnier, J., Kallel, L., and Schoenauer, M. (1999). Rigorous hitting times for binary mutations. *Evolutionary Computation* 7(2), 173–203.

Jägersküpper, J. (2003). Analysis of a simple evolutionary algorithm for minimization in Euclidian spaces. *Proc. of the 30th Internat. Colloq. on Automata, Languages, and Programming (ICALP 2003)*, LNCS 2719, 1068–1079.

Laumanns, M., Thiele, L., Zitzler, E., Welzl, E., and Deb, K. (2002). Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. *Proc. of the 7th Internat. Conf. on Parallel Problem Solving From Nature (PPSN VII)*, LNCS 2439, 44–53.

Motwani, R. and Raghavan, P. (1995). *Randomized Algorithms*. Cambridge University Press.

Rudolph, G. (1998a). Evolutionary search for minimal elements in partially ordered finite sets. *Proc. of the 7th Annual Conf. on Evolutionary Programming*, 345–353.

Rudolph, G. (1998b). On a Multi-Objective Evolutionary Algorithm and Its Convergence to the Pareto Set. *Proc. of the 5th IEEE Conf. on Evolutionary Computation*, 511–516.

Rudolph, G. (2001). Evolutionary search under partially ordered fitness sets. *Proc. of the Internat. NAISO Congress on Information Science Innovations (ISI 2001)*, 818–822.

Rudolph, G. and Agapie, A. (2000). Convergence properties of some multi-objective evolutionary algorithms. *Proc. of the 2000 Congress on Evolutionary Computation (CEC 2000)*, 1010–1016.

Scharnow, J., Tinnefeld, K., and Wegener, I. (2002). Fitness landscapes based on sorting and shortest paths problems. *Proc. of the 7th Conf. on Parallel Problem Solving from Nature (PPSN VII),* LNCS 2439, 54–63.

Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation* 2(3), 221–248.

Wegener, I. (2001). Theoretical aspects of evolutionary algorithms. *Proc. of the 28th Internat. Colloq. on Automata, Languages, and Programming (ICALP 2001)*, LNCS 2076, 64–78.

Wegener, I. and Witt, C. (2003). On the optimization of monotone polynomials by the (1+1) EA and randomized local search. *Proc. of the Genetic and Evolutionary Computation Conf. (GECCO 2003)*, LNCS 2723, 622–633.