Theoretical Computer Science 561 (2015) 24–36

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Population size matters: Rigorous runtime results for maximizing the hypervolume indicator

Anh Quang Nguyen [a,*], Andrew M. Sutton [b], Frank Neumann [a]

[a] Optimisation and Logistics, School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia
[b] Institut für Informatik, Friedrich-Schiller-Universität Jena, 07743 Jena, Germany

ARTICLE INFO

Article history:
Received 15 November 2013
Received in revised form 26 May 2014
Accepted 15 June 2014
Available online 19 June 2014

Keywords:
Evolutionary multi-objective optimization
Hypervolume indicator
Genetic programming
Theory
Runtime time analysis

ABSTRACT

Evolutionary multi-objective optimization is one of the most successful areas in the field of evolutionary computation. Using the hypervolume indicator to guide the search of evolutionary multi-objective algorithms has become very popular in recent years. We contribute to the theoretical understanding of these algorithms by carrying out rigorous runtime analyses. We consider multi-objective variants of the problems OneMax and LeadingOnes called OneMinMax and LOTZ, respectively, and investigate hypervolume-based algorithms with population sizes that do not allow coverage of the entire Pareto front. Our results show that LOTZ is easier to optimize than OneMinMax for hypervolume-based evolutionary multi-objective algorithms, which is contrary to the results on their single-objective variants and the well-studied $(1+1)$ EA. Furthermore, we study multi-objective genetic programming using the hypervolume indicator. We show that the classical ORDER problem is easy to optimize if the population size is large enough to cover the whole Pareto front and point out situations where a small population size leads to an exponential optimization time.

1. Introduction

Many real-world applications involve optimization problems with multiple objectives and evolutionary algorithms have become very popular for handling problems in the area of multi-objective optimization [9,10]. Evolutionary algorithms seem to be well-suited for these types of problems since, in multi-objective optimization, the goal is to compute a set of solutions that represent the trade-offs with respect to the given objective functions. Evolutionary multi-objective algorithms aim to compute a set of such trade-offs by iteratively evolving a population of solutions into a reasonable collection that represents a good set of trade-offs with respect to the given objective functions.

Hypervolume-based evolutionary algorithms have become very popular in recent years for multi-objective optimization [1,4,12,22]. These algorithms work with a fixed population size of μ and try to maximize the volume of space that is covered by the μ objective vectors corresponding to each individual of the population.

Despite their popularity, it is difficult to understand from a theoretical point of view how hypervolume-based evolutionary algorithms work. Many studies in recent years have concentrated on the optimal hypervolume distribution for a wide range of problems, or the cost of computing or approximating the value of the hypervolume indicator for a given set of μ points [5,6]. These studies tackled the difficult task of determining the configuration of the optimal distribution of the μ

* Corresponding author. Tel.: +61 425 631 956.
E-mail address: a.nguyen@adelaide.edu.au (A.Q. Nguyen).

http://dx.doi.org/10.1016/j.tcs.2014.06.023
0304-3975/

individuals without considering the computational costs to reach this goal. Other studies relate the optimal hypervolume distribution to the best achievable approximation ratio when using μ solutions to cover the Pareto front [3,7].

In this paper, we want to put forward the theoretical understanding of the optimization process of hypervolume-based evolutionary algorithms. We do this by carrying out rigorous runtime analyses of hypervolume-based evolutionary algorithms and point out, using popular example functions, when and why they are able to achieve an optimal hypervolume distribution in expected polynomial time. On the other hand, we show that even on very simple problems, hypervolume-based algorithms can have significant difficulties achieving the optimal configuration if the population size is not set correctly.

To our knowledge, the only article on the runtime analysis of hypervolume-based algorithms is the article by Brockhoff et al. [8]. We push forward the runtime analysis of hypervolume-based evolutionary algorithms by studying the algorithm called $(\mu+1)$ SIBEA introduced in [8] in order to extend these initial investigations.

We start by analyzing the OneMinMax problem introduced in [14], which is the multi-objective version of the famous OneMax problem. We show that as long as μ is large enough to cover the entire Pareto front, $(\mu+1)$ SIBEA computes the whole Pareto front in expected polynomial time. Furthermore, we point out that a smaller population size might lead to plateaus that are hard to leave in polynomial time. In particular, we show that there exists an initial population of size $\mu = O(\sqrt{n})$ such that the optimization time of $(\mu+1)$ SIBEA is exponential with probability very close to 1.

After having investigated OneMinMax, we turn our attention to the multi-objective problem LOTZ introduced in [18]. Extending the investigations of [8], we show that $(\mu+1)$ SIBEA optimizes LOTZ in expected polynomial time if $1 < \mu < n/3$. This shows that LOTZ is easier to optimize than OneMinMax by $(\mu+1)$ SIBEA for small μ.

This paper extends a preliminary conference version [20] by additional investigations into the field of tree-based genetic programming. Genetic programming has been shown to be a successful approach to evolve syntax trees that solve a given task [17]. It is frequently used in the area of symbolic regression [13]. We analyze multi-objective genetic programming algorithms using the hypervolume indicator and consider the multi-objective formulation of the ORDER problem [11,15]. Multi-objective genetic programming algorithms for this problem have already by analyzed with respect to their expected optimization time in [19]. We show that hypervolume-based multi-objective genetic programming algorithms solve the ORDER problem efficiently if the population size is large enough. For smaller population size we point out situations leading to an exponential optimization time.

The outline of the paper is as follows. In Section 2, we introduce the framework that is subject to our investigations. Section 3, studies the behavior of $(\mu+1)$ SIBEA on OneMinMax and these studies are extended to LOTZ in Section 4. Our results for multi-objective genetic programming using the hypervolume indicator are presented in Section 5. We finish with some concluding remarks.

2. Preliminaries

Let $\mathcal{X}$ be a finite domain. In multi-objective optimization, we work with a vector-valued fitness function $f : \mathcal{X} \to \mathbb{R}^m$ where $m \in \mathbb{N}$ and the fitness of an element $x \in \mathcal{X}$ is given by the vector $f(x) = (f_1(x), \ldots, f_m(x))$. Assuming we want to maximize each function f_i, we say $f(x) \geq f(x')$ if and only if $f_i(x) \geq f_i(x')$ for $x, x' \in \mathcal{X}$ and all $i \in \{1, \ldots, m\}$. We say a solution x dominates a solution x' if $f(x) \geq f(x')$ and $f(x) \neq f(x')$.

For a fitness function $f : \mathcal{X} \to \mathbb{R}^m$, we define

$$\mathscr{P} = \left\{x \in \mathcal{X} : f(x) \leq f\left(x'\right) \Longrightarrow f(x) = f\left(x'\right),\ \forall x' \in \mathcal{X}\right\}$$

as the Pareto set of f. We sometimes call a point $x \in \mathscr{P}$ *Pareto optimal*.

The *hypervolume indicator* is a set measure that identifies a set of elements in $\mathbb{R}^m$ (corresponding to images of elements in $\mathcal{X}$) with the volume of the dominated portion of the objective space. In particular, given a reference point $r \in \mathbb{R}^m$, the hypervolume indicator is defined on a set $A \subset \mathcal{X}$ as

$$I_H(A) = \lambda\left(\bigcup_{a \in A}\left[r_1, f_1(a)\right] \times \left[r_2, f_2(a)\right] \times \cdots \times \left[r_m, f_m(a)\right]\right)$$

where $\lambda(S)$ denotes the Lebesgue measure of a set S and $[r_1, f_1(a)] \times [r_2, f_2(a)] \times \cdots \times [r_m, f_m(a)]$ is the orthotope with r and $f(a)$ in opposite corners.

We define the *extreme points* of a problem as

$$\left\{x \in \mathscr{P},\ \exists i \in \{1, \ldots, m\} : f_i(x) \geq f_i\left(x'\right),\ \forall x' \in \mathcal{X}\right\}.$$

The extreme points are the points in $\mathscr{P}$ that maximize at least one objective function.

We study variants of $(\mu+1)$ SIBEA algorithm introduced by Brockhoff et al. [8] outlined in Algorithm 1. The algorithm uses a population of size μ and produces in each generation an offspring by mutation to create an intermediate population of $\mu+1$ individuals. The new parent population is then obtained in a steady-state fashion by deleting the individual that results in the smallest loss to the hypervolume indicator. This potential loss (called the contribution of an individual) is calculated as the difference between the indicator value of the population containing the individual, and the indicator value of a hypothetical population in which that individual is chosen for deletion.

Algorithm 1: $(\mu + 1)$ SIBEA.

Start with an initial population $P^{(1)}$ consisting of μ elements from $\mathcal{X}$.
repeat *forever*
  Select x from $P^{(t)}$ u.a.r.;
  $x' \leftarrow \text{mutate}(x)$;
  $P'^{(t)} \leftarrow P^{(t)} \cup \{x'\}$;
  foreach $x \in P'^{(t)}$ **do**
    $d(x; P'^{(t)}) \leftarrow I_H(P'^{(t)}) - I_H(P'^{(t)} \setminus \{x\})$;
  $z \leftarrow \arg\min_{x \in P'^{(t)}} d(x; P'^{(t)})$ with ties broken u.a.r.;
  $P^{(t+1)} \leftarrow P'^{(t)} \setminus \{z\}$;

A *trace* is an infinite stochastic sequence of μ-multisets

$$\left(P^{(1)}, P^{(2)}, \ldots, P^{(t)}, \ldots\right)$$

where $P^{(t)}$ denotes the population visited by $(\mu + 1)$ SIBEA in iteration t.

It will often be convenient to discuss how the properties of individual solutions change during the construction of an intermediate population $P'^{(t)}$ from a population $P^{(t)}$ in iteration t. To facilitate such discussion, we will employ the following notation scheme. We denote as x' the offspring produced by applying mutation to some individual in $P^{(t)}$. For each $x_i \in P^{(t)}$ we use the expression $x'_i \in P'^{(t)} \setminus \{x'\}$ to denote the same solution in the intermediate population directly after a mutation, but before undergoing truncation selection.

We are ultimately interested in the *optimization time* of $(\mu + 1)$ SIBEA, which is the random variable

$$T = \inf\left\{t \in \mathbb{N} : I_H\left(P^{(t)}\right) \text{ is maximal over } \mu\text{-multisets}\right\}.$$

In the following sections, we will derive asymptotic bounds for the expectation of T.

Given a population P, we define the hypervolume *contribution* of an individual $x \in P$ (with respect to P) as

$$d(x; P) = I_H(P) - I_H\left(P \setminus \{x\}\right).$$

Hence, $d(x; P)$ represents the volume of space that the point $x \in P$ uniquely contributes to the total hypervolume covered by the population P. When the population is clear from context, we will denote the contribution of x simply as $d(x)$.

We first consider multi-objective problems over length-n binary strings, that is $\mathcal{X} = \{0, 1\}^n$. For a point $x \in \{0, 1\}^n$, we will use square brackets to address individual elements of the bitstring, that is $x = (x[1], x[2], \ldots, x[n])$ so that $x[i] \in \{0, 1\}$ denotes the i-th bit in the string x. The mutation operator that we consider differs slightly from the variants of $(\mu+1)$ SIBEA initially proposed by [8]. In particular, we will study only a local, single-point mutation operator, in which a single bit, chosen uniformly at random, is flipped at each mutation step. In this way, the algorithms we consider are more similar to indicator-based randomized local search. This single-point mutation operator is defined as follows.

Function mutate(x).

$x' \leftarrow x$;
Choose $i \in \{1, \ldots, n\}$ u.a.r.;
$x'[i] \leftarrow 1 - x'[i]$;
return x';

In the case of bicriteria optimization, we will be interested in the dynamics of the *intervals* between points on one fitness function, say f_1. To do so, we label a population P with respect to its ordering on f_1, that is $P = \{x_1, x_2, \ldots, x_\mu\}$ such that $f_1(x_i) \leq f_1(x_{i+1})$ and define an interval function as follows.

Definition 1. Given a labeled population P, the *interval function* $\iota\colon P \setminus \{x_1\} \to \mathbb{R}$ measures the intervals between the projection of the points onto the f_1 axis (see Fig. 1) as follows.

$$\iota(x_i) = f_1(x_i) - f_1(x_{i-1}) \quad \text{for all } i \in \{2, 3, \ldots, \mu\}.$$

In the remainder of the paper we rigorously prove theorems about the optimization time of $(\mu + 1)$ SIBEA on various multi-objective problems. In each case we provide asymptotic bounds on the expected time to maximize the hypervolume indicator, and these bounds are functions of both problem size and population size μ. It is worth remarking here that a change in μ has the effect of changing the properties of the hypervolume indicator and hence the ability of the algorithm to cover the Pareto front. For a given μ, we always consider the optimal solution to be the configuration that results in the highest hypervolume indicator value that could possibly be achieved by any population of size μ.

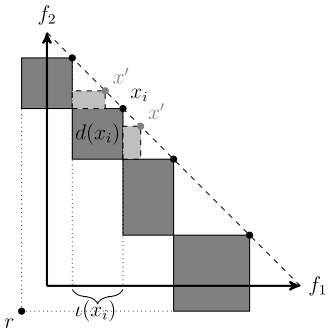

Fig. 1. Shaded areas denote contribution $d(x_i)$ of each individual x_i to the hypervolume indicator.

3. OneMinMax

The OneMinMax problem, originally introduced by Giel and Lehre [14], is defined as the bicriteria optimization problem over the set of binary strings as $f : \{0,1\}^n \to \mathbb{N} \times \mathbb{N}$ with

$$f(x) = \big(f_1(x), f_2(x)\big), \quad \text{with}$$
$$f_1(x) = |x|_0 = n - |x|_1, \text{ and}$$
$$f_2(x) = |x|_1 = \sum_{i=1}^{n} x[i].$$

The goal is to maximize both objectives. In OneMinMax, every solution belongs to $\mathscr{P}$. Denoting as $f[\mathscr{P}]$ the image of $\mathscr{P}$ under f in objective space, it is clear that $|f[\mathscr{P}]| = n+1$, since each string with k ones ($n-k$ zeros) is mapped to a unique, non-dominated point in objective space. We define the reference point to be $r = (-n^2, -n^2)$ such that the extreme points always have the highest contributions. We are interested in the expected time for $(\mu+1)$ SIBEA to arrange the population on the Pareto front so that the hypervolume indicator is maximized.

Theorem 1. *If $\mu \geq n+1$, the expected optimization time of $(\mu+1)$ SIBEA on* OneMinMax *is $O(\mu n \log n)$.*

Proof. Since every solution belongs to $\mathscr{P}$ and $\mu \geq n+1$, solutions are never lost during a run. The algorithm reaches the optimal hypervolume when the population covers the whole Pareto front. In the case of OneMinMax, a population P covers the whole Pareto front when, for every $i \in \{0,1,\ldots,n\}$, there exists an $x \in P$ such that $|x|_1 = i$ and $|x|_0 = n - i$. Thus it suffices to bound the expected time it takes for $(\mu+1)$ SIBEA to generate a solutions for all missing values of $|\cdot|_1$.

Suppose t is a generation in which $I_H(P^{(t)})$ is suboptimal. In such a generation, the Pareto front is not yet covered. Hence, there exists at least one individual $x \in P^{(t)}$ such that at least one of the following is true.

1. $\nexists x' \in P^{(t)}, |x'|_1 = |x|_1 + 1$.
2. $\nexists x' \in P^{(t)}, |x'|_1 = |x|_1 - 1$.

It is thus possible to gain a missing $|\cdot|_1$ value by mutating x by flipping either a one bit or a zero bit. Without loss of generality, suppose there are no solutions $x' \in P^{(t)}$ with $|x'|_1 = |x|_1 + 1$ and $|x|_1 < n$ (otherwise, a symmetric argument holds for $|x'|_1 = |x|_1 - 1$ and $|x|_1 > 0$). In this case, a mutation operation creates a new Pareto optimal solution if x is selected for mutation, and one of the zero bits in x is flipped to produce a new solution x'' where $|x''|_1 = |x'|_1$. The probability of this event is at least $|x|_0/(\mu n) = (n - |x|_1)/(\mu n)$. The total expected waiting time to generate all missing $|\cdot|_1$ values is thus bounded by

$$\sum_{i=0}^{n} \frac{\mu n}{n-i} = \mu n \sum_{i=0}^{n} \frac{1}{i} = O(\mu n \log n).$$

The same argument holds in the case of missing values for $|x|_1 - 1$, and the claimed bound on the expected time to cover the Pareto front follows. □

Theorem 1 shows that, so long as the population is sufficiently large, the optimization time of $(\mu+1)$ SIBEA on OneMinMax is bounded by a polynomial in μ and n. In contrast, we will soon show (in Theorem 3) that if the population size is too small, the optimization time can become exponential in n. Before stating and proving this result, we will need to develop a few technical concepts.

On OneMinMax for strings of length n, a population P is distributed along the line $x + y = n$ in the xy-plane. We label the population $P = \{x_1, x_2, \ldots, x_\mu\}$ such that $|x_i|_0 \leq |x_{i+1}|_0$. The following lemma states that, after a polynomially-bounded number of iterations of $(\mu + 1)$ SIBEA, the population contains the extreme points $(00\cdots0)$ and $(11\cdots1)$.

Lemma 1. *If $\mu \geq 2$, the expected time until the two extreme points are contained in the population is $O(\mu n \log n)$.*

Proof. In OneMinMax, the two extreme points are the points $(00\cdots0)$ and $(11\cdots1)$. In every iteration during the run of $(\mu + 1)$ SIBEA, the value corresponding to the highest $|\cdot|_1$ value in the population never decreases (since removing this point results in a population with a strictly smaller hypervolume indicator value).

In iteration t, the individual $x \in P^{(t)}$ with the highest $|\cdot|_1$ value is selected with probability $1/\mu$, and a zero bit is selected with probability k/n where k is the number of zero bits in x. The highest $|\cdot|_1$ value can be increased at most n times until $(11\cdots1)$ enters the population. A symmetric argument holds for the $|\cdot|_0$ value, and the expected time to until both extreme points enter the population is upper bounded by:

$$2\sum_{k=1}^{n}\left(\frac{1}{\mu}\frac{k}{n}\right)^{-1} = 2\mu n\sum_{k=1}^{n}\frac{1}{k} = O(\mu n \log n). \qquad \square$$

Without loss of generality, in order to simplify the analysis we will assume for the remainder of the paper that n is divisible by $\mu - 1$. We remark here that our results still hold in the case that $\mu - 1$ does not divide n evenly, but this requires some extra attention to technicalities.

The *optimal μ-distribution* for a Pareto front $\mathscr{P}$ is the set of μ points on the front that maximize the hypervolume indicator. The following theorem is due to Auger et al. [2, Theorem 7].

Theorem 2. *If the Pareto front is a (connected) line, the optimal μ-distribution with respect to the hypervolume indicator is such that the distance is the same between all neighboring solutions.*

In other words, assuming $2 < \mu < n + 1$ and $\mu - 1$ divides n evenly, in the optimal μ-distribution, the intervals $\iota(x_i) = \iota(x_j) = n/(\mu - 1)$ for all $i, j \in \{2, \ldots, \mu\}$.

Since OneMinMax has a linear front, the conditions for the theorem are satisfied.

Proposition 1. *Suppose the extreme points are contained in P. Let x' be produced from $x_i \in P$ by flipping a single bit. Then either,*

$$d(x') = |x_{i+1}|_0 - |x_i|_0 - 1 = \iota(x_{i+1}) - 1, \quad \text{or},$$
$$d(x') = |x_i|_0 - |x_{i-1}|_0 - 1 = \iota(x_i) - 1.$$

Note that the assumption on the membership of extreme points implies $|x_1|_0 \leq |x'|_0 \leq |x_\mu|_0$ and at least one of the above statements is always well-defined.

An illustration of Proposition 1 is given in Fig. 1.

Proposition 2. *The contribution of a solution $x_i \in P \setminus \{x_1, x_\mu\}$ is calculated by*

$$d(x_i) = \big(|x_i|_0 - |x_{i-1}|_0\big) \times \big(|x_{i+1}|_1 - |x_i|_1\big),$$

and since $|x_i|_1 = n - |x_i|_0$,

$$d(x_i) = \iota(x_i)\iota(x_{i+1}).$$

Lemma 2. *Given a population $P^{(t)}$ on* OneMinMax, *$(\mu + 1)$ SIBEA can produce a new population $P^{(t+1)}$ with $I_H(P^{(t+1)}) > I_H(P^{(t)})$ if and only if one of the following conditions is satisfied:*

1. $\exists x_i \in P^{(t)} \setminus \{x_1\} : |\iota(x_i) - \iota(x_{i+1})| \geq 2$.
2. $\exists x_i, x_j \in P^{(t)} \setminus \{x_1\} : \iota(x_i) > d(x_j) + 1$.

Specifically, we show that at least one of these conditions is necessary and sufficient for $(\mu + 1)$ SIBEA to produce an offspring x' in generation t such that

$$\exists z \in P^{(t)} : I_H\big(\big(P^{(t)} \setminus \{z\}\big) \cup \{x'\}\big) > I_H\big(P^{(t)}\big)$$

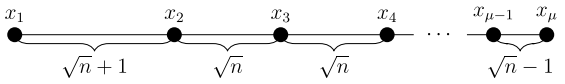

Fig. 2. The intervals corresponding to population P^w for OneMinMax.

Proof. For the first condition, without loss of generality, assume that $\iota(x_{i+1}) = \iota(x_i) + k$ for $k \geq 2$. In order for a strict improvement to occur, a new solution x' can be generated by flipping a one bit in x_i to a zero. We calculate the new intervals in the intermediate population $P'^{(t)} \leftarrow P^{(t)} \cup \{x'\}$ as $\iota(x'_i) = \iota(x_i)$, $\iota(x') = 1$,[1] and $\iota(x'_{i+1}) = \iota(x_{i+1}) - 1$. It follows that $d(x'_i) = \iota(x'_i) = \iota(x_i) < \iota(x_i) + k - 1 = \iota(x_{i+1}) - 1 = d(x')$. Hence replacing x'_i with x' results in a strictly higher hypervolume indicator.

For the second condition, assume that there are two solutions x_i, x_j such that $\iota(x_i) > d(x_j) + 1$. Let x' be a new solution produced by mutating x_i or x_{i-1}. According to Proposition 1, $d(x') = |x_i|_0 - |x_{i-1}|_0 - 1 = \iota(x_i) - 1 > d(x_j)$.

Therefore, if one of the above conditions is satisfied, there always exists a solution z in the intermediate population such that

$$I_H\left(P^{(t+1)} \leftarrow P'^{(t)} \setminus \{x'\}\right) < I_H\left(P^{(t+1)} \leftarrow P'^{(t)} \setminus \{z\}\right)$$

On the other hand, suppose none of the above conditions hold. In this case, for all solutions $x_i, x_j \in P^{(t)} \setminus \{x_1\}$, $|\iota(x_i) - \iota(x_{i+1})| \leq 1$ and $\iota(x_i) \leq d(x_j) + 1$. Assume that a solution $x_k \in P^{(t)}$ is selected for mutation and let x' be the new solution obtained by mutating x_k. Again, by Proposition 1, either $d(x') = |x_{k+1}|_0 - |x_k|_0 - 1$ or $d(x') = |x_k|_0 - |x_{k-1}|_0 - 1$ and we make the following case distinction.

Case 1. If $d(x') = |x_{k+1}|_0 - |x_k|_0 - 1 = \iota(x_{k+1}) - 1$, then only the value of $\iota(x_{k+1})$ changes, which leads to $d(x'_k) = \iota(x_k)$, $d(x'_{k+1}) = (\iota(x_{k+1}) - 1) \times \iota(x_{k+2})$ while the contribution of other solutions in the population remain the same. Obviously, $d(x') \leq d(x'_{k+1})$ and $d(x') \leq d(x'_k)$ from the first counter conditions. Furthermore, based on the second counter condition, $d(x') \leq d(x'_j)$ for all $j \neq k$ and $j \neq k+1$.

Case 2. If $d(x') = |x_k|_0 - |x_{k-1}|_0 - 1 = \iota(x_k) - 1$, then only the value of $\iota(x_k)$ changes, which leads to $d(x'_k) = \iota(x_{k+1})$, $d(x'_{k-1}) = (\iota(x_k) - 1) \times \iota(x_{k-1})$ while the contribution of other solutions in the population remain the same. Using a similar argument, $d(x') \leq d(x'_j)$ for all $1 \leq j \leq \mu$.

Therefore, if none of the conditions are satisfied, for all solutions x_j in the population,

$$I_H\left(P^{(t+1)} \leftarrow P'^{(t)} \setminus \{x_j\}\right) \leq I_H\left(P^{(t+1)} \leftarrow P'^{(t)} \setminus \{x'\}\right).$$

This completes the proof. □

We define an initial population P^w in which (1) $\mu = \sqrt{n} + 1$, (2) the extreme points already exist in P^w, and (3) the intervals are as follows.

$$\iota(x_i) = \begin{cases} \sqrt{n} + 1 & \text{if } i = 2; \\ \sqrt{n} - 1 & \text{if } i = \mu; \\ \sqrt{n} & \text{if } i \in \{3, \ldots, \mu - 1\}. \end{cases}$$

The set of intervals of P^w with respect to $f_1 = |\cdot|_0$ are illustrated in Fig. 2.

We now show that with probability approaching one exponentially fast $(\mu + 1)$ SIBEA requires exponential time to reach the optimal population state from P^w.

Theorem 3. *Starting with the initial population P^w where $\mu = \sqrt{n} + 1$, the expected optimization time of $(\mu + 1)$ SIBEA on* OneMinMax *is exponential with probability $1 - 2^{-\Omega(n)}$.*

Proof. By Theorem 2, the population is optimal when $\iota(x_i) = \sqrt{n}$ for all $x_i \in P^{(t)} \setminus \{x_1\}$. During the run, we call an interval *large* if $\iota(x_i) = \sqrt{n} + 1$. Similarly, we call the interval *small* if $\iota(x_i) = \sqrt{n} - 1$. At the beginning of the above configuration, the interval belonging to x_2 is large, and the interval belonging to x_μ is small.

In this configuration, an offspring is only accepted if it results in moving the large interval or the small interval in either direction by flipping one of the bits in the solutions that bound the intervals.

No other mutation will be accepted since it would result in an offspring with a strictly smaller contribution. To reach the optimum, the large interval and the small interval must meet, i.e., $\iota(x_i) = \sqrt{n} + 1$ and $\iota(x_{i+1}) = \sqrt{n} - 1$ for some $i \in \{2, \ldots, \mu\}$. From this state, only a single mutation is possible to reach the optimum.

[1] Note that here we are abusing the ι notation slightly. The μ intervals in the intermediate population should be understood in terms of their relative positions along the f_1-axis. This may be different than their canonical subscripts due to the addition of x'.

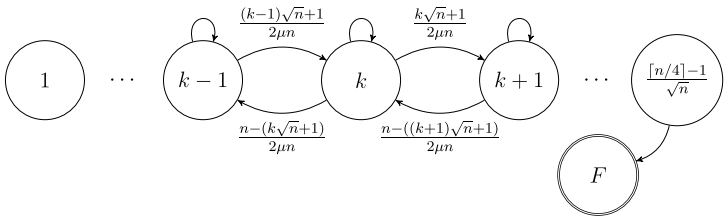

Fig. 3. Markov chain for interval dynamics for $(\mu+1)$ SIBEA on OneMinMax.

Let T be the random variable corresponding to the optimization time of $(\mu+1)$ SIBEA. We consider the position of the large and small intervals during all times $t < T$. Let $a = a(t)$ be the index at time t such that $\iota(x_a) = \sqrt{n}+1$ and let $b = b(t)$ be the index at time t such that $\iota(x_{b+1}) = \sqrt{n}-1$. The optimal solution is only reachable from the state in which $a = b$. Before this occurs, at least one of the following conditions must hold at some time $t < T$. (1) $|x_a|_0 \geq \frac{n}{4}$, or (2) $|x_b|_0 \leq \frac{n}{4}$.

We now prove that either of these conditions only occur after exponential time with probability approaching one exponentially fast. For the first condition, note that probability of a mutation that moves the large interval to the right is $|x_a|_0/(2\mu n)$. Here, $1/\mu$ is the probability of choosing the correct individual, x_a from the population, $|x_a|_0/n$ is the probability of flipping one of zero bits in the current solution and $1/2$ is the probability of keeping the new solution.

Meanwhile, the probability of moving the large interval to the left (if possible) is $|x_{a-1}|_1/(2\mu n)$ where $1/\mu$ is the probability of choosing the correct solution x_{a-1} from the population, $|x_{a-1}|_1/n$ is the probability of flipping one of the one bits in the current solution and $1/2$ is the probability of keeping the new solution.

Let X_t be the set of variables describing a Markov process over a finite state space $S = \{1, 2, \ldots, (\lceil n/4\rceil - 1)/\sqrt{n}\} \cup F$ where the k-th state corresponds to the event in which $|x_{a(t)}|_0 = k\sqrt{n}+1$, that is, the large interval has moved into position k. The chain is absorbed into a final accepting state F if the small interval meets it (we later show this also takes exponential time). Letting T denote the time when the intervals first meet, for all $0 \leq t < T$, $X_{t+1} - X_t \in \{-1, 0, 1\}$. Using the above transition probabilities (illustrated in Fig. 3), for all $0 < t < T$ and $1 \leq X_t \leq (\lceil n/4\rceil - 1)/\sqrt{n}$,

$$\Pr(X_{t+1} - X_t = -1 \mid X_t) \geq \delta \cdot \Pr(X_{t+1} - X_t = 1 \mid X_t),$$

where $\delta \geq (n - n/4)/(n/4) = 3$. By the Local Gambler's Ruin Theorem [16], the time until $a(t)$ is large enough so that $|x_{a(t)}|_0 \geq n/4$ conditioned on the event that it has not yet met the small interval is at least $\delta^{1/3\cdot(1-1/(4\mu))\cdot n} \geq 3^{\Omega(n)}$ with probability $1 - 2^{-\Omega(n)}$. A symmetric argument shows that, with high probability, it requires exponential time for $b(t)$ to reach $|x_{b(t)}|_0 \leq n/4$. Since both conditions require exponential time with probability $1 - 2^{-\Omega(n)}$, the claim follows. □

4. LOTZ

The *leading ones, trailing zeros* (LOTZ) problem is a bicriteria optimization problem over binary strings first introduced by Laumanns et al. [18], defined as $\mathrm{LOTZ} : \{0,1\}^n \to \mathbb{N}^2$ where

$$\mathrm{LOTZ}(x) = \big(f_1(x), f_2(x)\big) \quad \text{with}$$
$$f_1(x) = \mathrm{LO}(x) = \sum_{i=1}^{n}\prod_{j=1}^{i} x[j], \text{ and}$$
$$f_2(x) = \mathrm{TZ}(x) = \sum_{i=1}^{n}\prod_{j=i}^{n}\big(1 - x[j]\big).$$

Similar to OneMinMax, we set the reference point as $r = (-n^2, -n^2)$ and label the population $P = \{x_1, x_2, \ldots, x_\mu\}$ such that $\mathrm{LO}(x_i) \leq \mathrm{LO}(x_{i+1})$.

Since the Pareto front of LOTZ is linear, Theorem 2 holds. Moreover, as long as all points are on the Pareto front, Propositions 1 and 2 hold (using the requisite objective function) and Lemma 2 carries over to LOTZ as well. The following theorem follows directly from Brockhoff, Friedrich and Neumann [8, Theorem 2].

Theorem 4. *If $\mu \geq n+1$, the expected optimization time of $(\mu+1)$ SIBEA on* LOTZ *is $O(\mu n^2)$.*

Though the proof of Theorem 4 is carried out for a more general version of $(\mu+1)$ SIBEA, the claim holds also for our choice of mutation operator, which is only slightly easier to control.

Theorem 5. *Let $1 < \mu < n+1$. The expected time for $(\mu+1)$ SIBEA to reach a population that contains only Pareto optimal solutions and solutions for the objective vectors $(n, 0)$ and $(0, n)$ is $O(\mu^2 n^2)$.*

Proof. In any point $x \notin \mathscr{P}$, a 0 must appear before a 1 in x. Thus, $x = 1^a 0\{0,1\}^b 10^c$ for $a, b, c \geq 0$. A mutation is only accepted if the 0 in position $a+1$ is flipped, or the 1 in position $a+b+2$ is flipped. In either case, the resulting offspring must have a strictly higher LO (TZ) value (respectively), so the highest LO (TZ) value in the population never decreases.

Let $x \in P^{(t)}$ be the point in the population with the highest LO value in iteration t. The probability that x is selected for mutation, and the correct bit is flipped to increase the LO value is bounded below by $1/(\mu n)$. This event can occur a maximum of n times until the extreme point with the highest LO value is in the population. It follows that the expected time until the extreme point with the highest LO value is $O(\mu n^2)$. A symmetric argument holds for the extreme point with the highest TZ value. Thus the expected time until both extreme points are in the population is $O(\mu n^2)$.

During the runtime of the algorithm, the count of Pareto optimal solutions in the population never decreases since each accepted mutation on a non-Pareto optimal solution will generate an offspring that either dominates or is dominated by its parent. The number of Pareto optimal solutions can increase when there exists a dominated solution in the population, or the new Pareto optimal solution dominates one of the non-Pareto, non-dominated solutions.

In the first case, where there is at least a dominated solution, there exists a point $x \in P^{(t)} \cap \mathscr{P}$ with a Hamming neighbor x' such that $x' \in \mathscr{P}$ but $x' \notin P^{(t)}$. The probability of creating x' by applying a mutation to x is $1/(\mu n)$. Such a mutation must increase the number of Pareto optimal solutions in the population by one.

In the second case, where there is no dominated solution in the population, a new Pareto optimal solution that satisfies the previous conditions can be obtained by moving a non-dominated solution toward the Pareto front. A non-dominated solution can be moved closer to the Pareto front by first selecting it for mutation and then mutating the correct bit. This event occurs with probability $1/(\mu n)$. Since there are at most n bits that need to be flipped, the expected time to increase the number of Pareto optimal solutions in the population is $O(\mu n^2)$. Hence, the expected waiting time until all μ individuals are Pareto optimal is bounded above by $O(\mu^2 n^2)$ and the claimed bound follows. □

We now consider some conditions on a given population that imply the existence of mutations that can improve the hypervolume indicator.

Theorem 6. *Let $1 < \mu < \frac{n}{3}$ and let $P^{(t)}$ be the population in iteration t of $(\mu+1)$ SIBEA on* LOTZ. *If $P^{(t)} \subseteq \mathscr{P}$ and there is no single mutation that results in a population with a strictly higher hypervolume indicator, then $\iota(x_i) \geq 2$ for all $x_i \in P^{(t)} \setminus \{x_1\}$.*

Proof. Since $1 < \mu < \frac{n}{3}$, if $P^{(t)}$ is optimal, then for all $x_i \in P^{(t)} \setminus \{x_1\}$, $\iota(x_i) \geq 3$. Suppose $P^{(t)}$ is not optimal. If there exists a solution $x_i \in P^{(t)} \setminus \{x_1\}$ with such that $\iota(x_i) = 1$, then there must also be a solution x_j where $\iota(x_j) \geq 4$. If a strict improvement cannot be made in a single mutation, then the first condition of Lemma 2 tells us that $|\iota(x_i) - \iota(x_{i+1})| \leq 1$, which implies that $0 \leq \iota(x_{i+1}) \leq 2$ (recall here that Lemma 2 carries over to LOTZ when all points in the population are in $\mathscr{P}$). Therefore, $d(x_i) = \iota(x_i) \times \iota(x_{i+1}) \leq 2 < \iota(x_j) - 1$, and the second condition of Lemma 2 is satisfied. This contradicts the assumption that no improving mutation is possible. □

We define a *neutral move* to be the transformation of a population $P^{(t)}$ via mutation and selection to a population $P^{(t+1)}$ such that $I_H(P^{(t)}) = I_H(P^{(t+1)})$.

Lemma 3. *Let $1 < \mu < \frac{n}{3}$ and let $P^{(t)}$ be the population in iteration t of $(\mu+1)$ SIBEA on* LOTZ. *If $P^{(t)} \subseteq \mathscr{P}$ and there exists a solution $x_k \in P^{(t)} \setminus \{x_1\}$ such that $\iota(x_k) = 1$, then using only neutral moves, it is not possible for the population to enter a state in which a strict improvement cannot occur in one mutation.*

Proof. Suppose for contradiction that it is possible for a neutral move to result in a population $P^{(t+1)}$ from which a strict improvement cannot occur. By Theorem 6, for all $x_i \in P^{(t+1)} \setminus \{x_1\}$, $\iota(x_i) \geq 2$. This means that to produce $P^{(t+1)}$, either x'_k or x'_{k-1} must be removed from the intermediate population $P'^{(t)}$. Furthermore, when a solution in $P^{(t)}$ is mutated to produce the intermediate population, two new intervals are produced with at least one of them equal to 1. The only configuration in which two unit intervals can be removed from the intermediate population is when the unit intervals are adjacent on the Pareto front and the solution between them is subsequently removed. Thus to produce such an intermediate population, it follows that x_k, x_{k+1} or x_{k-1} must be mutated.

In the first case, when x_k is mutated, let x' be the new solution. We have $\iota(x') = 1$, $\iota(x'_{k+1}) = \iota(x_{k+1}) - 1 \geq 1$ so that $d(x'_k) = 1 \leq d(x')$. Because this is a neutral move, $d(x') = d(x'_k) = 1$, which means $\iota(x'_{k+1}) = 1 = \iota(x') = \iota(x'_k)$.

In the second case, without loss of generality, assume that x_{k-1} is mutated to produce a new solution x'. Using the same argument, $d(x') = d(x'_{k-1}) = 1$, which implies that $\iota(x_{k-1}) = 2$ and after mutation, $\iota(x'_k) = \iota(x'_{k-1}) = \iota(x') = 1$.

Thus, as long as there is a solution x_k in $P^{(t)} \setminus \{x_1\}$ such that $\iota(x_k) = 1$, any neutral move produces a new population $P^{(t+1)}$ such that there must exist some $x_i \in P^{(t+1)} \setminus \{x_1\}$ with $\iota(x_i) = 1$. However, by the contrapositive of Theorem 6, a strict improvement is possible in $P^{(t+1)}$. □

If no strict improvement is possible, then $(\mu+1)$ SIBEA working on LOTZ is on a plateau. To analyze its behavior in this situation, we appeal to the following result for linear Markov chains.

Lemma 4. *Let $N \in \mathbb{N}$ and let $\{X_t : t \in \mathbb{N}\}$ be a linear Markov chain on the state space $\{0, 1, \ldots, N\}$ specified as follows. For $0 < p < 1$,*

1. *if $X_t = N$ then $X_{t+1} = N - 1$ with probability at least p, otherwise $X_{t+1} = N$.*
2. *if $X_t = 0$, then $X_{t+1} = 0$.*
3. *otherwise,*

$$X_{t+1} = \begin{cases} X_t + 1 & \text{with probability } p/2, \\ X_t - 1 & \text{with probability } p/2, \\ X_t & \text{with probability } 1 - p. \end{cases}$$

All other state transitions have probability zero. Define the random variable $T := \inf\{t : X_t = 0\}$. Then $\mathrm{E}(T) = O(N^2/p)$.

Proof. Denote the filtration $\mathcal{F}_t = (X_1, \ldots, X_t)$ and let $\Delta_t = X_{t+1} - X_t$. Clearly $\mathrm{E}(\Delta_t \mid \mathcal{F}_t) \leq 0$ and $\mathrm{E}(\Delta_t^2 \mid \mathcal{F}_t) \geq p$. Define the stochastic process $Y_t = X_t^2 - 2NX_t - tp$. The process Y_t is a submartingale with respect to X_t since

$$\begin{aligned} \mathrm{E}(Y_{t+1} \mid \mathcal{F}_t) &= \mathrm{E}\big((X_t + \Delta_t)^2 - 2N(X_t + \Delta_t) - (t+1)p \mid \mathcal{F}_t\big) \\ &= X_t^2 - 2NX_t - tp + \big(\mathrm{E}\big(\Delta_t^2 \mid \mathcal{F}_t\big) - p\big) + \big((2X_t - 2N)\mathrm{E}(\Delta_t \mid \mathcal{F}_t)\big) \\ &\geq Y_t. \end{aligned}$$

Then by the Optional Stopping Theorem [21], $\mathrm{E}(Y_T) \geq \mathrm{E}(Y_0)$ so $\mathrm{E}(X_T^2) - 2N\mathrm{E}(X_T) - p\mathrm{E}(T) \geq \mathrm{E}(X_0^2) - 2N\mathrm{E}(X_0)$. Because $X_T = 0$, it follows that

$$\mathrm{E}(T) \leq \frac{2N\mathrm{E}(X_0) - \mathrm{E}(X_0^2)}{p},$$

and the claim holds since $0 \leq X_0 \leq N$. □

Lemma 5. *Let $1 < \mu < \frac{n}{3}$ and let $P^{(t)}$ be the population in iteration t of $(\mu+1)$ SIBEA on* LOTZ. *If $P^{(t)} \subseteq \mathscr{P}$ and no strict improvement is possible, then the expected time until $(\mu+1)$ SIBEA leaves the plateau and achieves a strict improvement is $O(\mu^3 n)$.*

Proof. Suppose no strict improvement is possible from $P^{(t)}$. By Lemma 2, $\iota(x_i) \leq d(x_j) + 1$ for all $x_i, x_j \in P^{(t)} \setminus \{x_1\}$. In this situation, the solutions in the population need to be shifted around in order for a strict improvement to be possible. $(\mu+1)$ SIBEA must move first along a plateau before encountering an improving move.

By Theorem 6, for all $x_i \in P^{(t)} \setminus \{x_1\}$, we have $\iota(x_i) \geq 2$, which implies that $d(x_i) \geq 4$. If there exist two solutions $x_i, x_j \in P^{(t)} \setminus \{x_1\}$ such that $\iota(x_i) = d(x_j) + 1 \geq 5$, a strict improvement can be made after a single neutral move. In this neutral move, either x_i or x_{i-1} is selected from the population with probability $1/\mu$ and a correct bit is flipped with probability $1/n$ to generate a new solution x' with contribution $d(x') = \iota(x_i) - 1 \geq 4$. Since there is at least one solution $x'' \in P^{(t)}$ that has the same contribution as x', it follows that x' replaces x'' to create population $P^{(t+1)}$ with probability at least $1/2$.

If this mutation is accepted, it is not possible for a subsequent mutation to revert to the previous population state. This is because a single mutation always results in two new intervals in which at least one of them is equal to one. Since we supposed that $P^{(t)}$ had no such interval, the claim follows. Moreover, after this step, there are two adjacent intervals with values 1 and $\iota(x_i) - 1$ in the population, and a mutation that strictly improves the hypervolume indicator is possible. Again, such a mutation occurs with probability $1/(\mu n)$. Finally, we point out that after this initial step, by Lemma 3, it is not possible to enter a state in which a strict improvement is impossible.

Otherwise, we have $\iota(x_i) < d(x_j) + 1$ for all $1 < i, j \leq \mu$. Let x_i, x_j with $i < j$ be the two closest solutions such that $|\iota(x_i) - \iota(x_j)| \geq 2$. The following two conditions are guaranteed. (1) $\iota(x_{i+1}) = \iota(x_{i+2}) = \cdots = \iota(x_{j-1}) = (\iota(x_i) + \iota(x_j))/2 = \alpha$. Without loss of generality, assume that $\iota(x_i) < \iota(x_j)$, for a solution x_k $(i < k < j)$, if $\iota(x_k) > \alpha$ or $\iota(x_k) < \alpha$ then x_k, x_j and x_i, x_k become the closest solutions accordingly. (2) $|\iota(x_i) - \iota(x_j)| = 2$. Otherwise, $|\iota(x_i) - \iota(x_j)| > 2$ implies that $|\iota(x_i) - \alpha| = |\iota(x_j) - \alpha| > 2$. This contradicts the supposition that x_i and x_j are the closest solutions that satisfy the condition.

According to Lemma 2, in order for a strict improvement to be possible, $\iota(x_i)$ and $\iota(x_j)$ must meet, implying that $j = i+1$. In one step, the distance between x_i and x_j can be changed by either moving x_i and x_j closer or further. The probability of moving x_i toward x_j is $1/(2\mu n)$. Here, the probability $1/2$ comes from the fact that after flipping a correct bit, there are at most two solutions in the population with the same contribution. Similarly, the probability of moving x_i further away from x_j is $1/(2\mu n)$, because we need only to flip the correct bit in the correct individual and maintain it in the population. Finally, by the same argument, the probability of moving x_j toward (further from) x_i is $1/(2\mu n)$. It follows that the probability that x_i and x_j are moved closer (further) is $2/(2\mu n) = 1/(\mu n)$.

After x_i and x_j become adjacent on the Pareto front, a strict improvement can be performed with probability $1/(\mu n)$. The process of shifting the distance between x_i and x_j is illustrated in the Markov chain in Fig. 4. In this Markov chain, there are $\mu + 1$ states that each correspond to the distance between x_i and x_j, and a final absorbing state reachable from the zero-distance state that corresponds to an improving move.

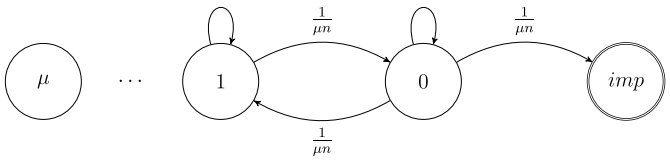

Fig. 4. Markov chain for interval dynamics for $(\mu+1)$ SIBEA on LOTZ. State *imp* represents a strictly improving move.

Excluding transition probabilities that are zero, the structure of the state transition graph is a simple path and the transition probabilities between neighboring states in the path (excluding the absorbing state) are all equal to $1/(\mu n)$. This results in a linear Markov chain with $N = \mu + 1$ and $p = 2/(\mu n)$ and the bound follows from Lemma 4. □

Finally, we can state the main theorem for LOTZ, which shows that the maximum hypervolume is obtained in expected polynomial time if $1 < \mu < \frac{n}{3}$ holds.

Theorem 7. *If* $1 < \mu < \frac{n}{3}$, *the expected optimization time of* $(\mu+1)$ *SIBEA on* LOTZ *is* $O(\mu^3 n^3)$.

Proof. If a strict improvement is possible, the expected waiting time until the algorithm discovers that improvement is bounded by $O(\mu n)$. There are at most $O(n^2)$ strict improvements, which implies that the algorithm needs at most $O(\mu n^3)$ to reach the maximum population hypervolume, as long as the algorithm never visits a state from which there are no strict improvements.

On the other hand, if the algorithm reaches a state from which a strict improvement is not possible, it must then deal with a plateau. In this case, by Lemma 5, the waiting time until the algorithm leaves the plateau and discovers a strict improvement is $O(\mu^3 n)$. The total number of improvements with respect to the hypervolume is $O(n^2)$, which implies that $(\mu+1)$ SIBEA spends at most $O(\mu^3 n^3)$ steps in such situations.

The expected optimization time is the total expected time spent in each of these two scenarios. Since this is dominated by $O(\mu^3 n^3)$, the proof is complete. □

5. Multi-objective genetic programming

We now shift our focus from binary strings and consider the domain of tree-based genetic programming (GP). Genetic programming is a class of evolutionary algorithms that evolve executable structures such as computer programs for some specific task [17]. In this particular application, candidate solutions are syntax trees whose nodes are a set of primitives comprised of a set of function nodes and a set of terminal symbols.

We consider $(\mu+1)$ SIBEA (see Algorithm 1), but the search space consists now of syntax trees. The inner nodes of each syntax tree are labeled by function symbols from a function set F and the leaves are labeled by terminals coming from a set L. For a given syntax tree x the mutation operator should produce a new syntax tree by applying local changes to x. To do so, we employ the HVL-prime operator [11]. HVL-prime mutation produces a new tree by making changes to the original tree via three basic operators: insertion, substitution, and deletion. We modify Algorithm 1 by changing the mutate function call to hvl-mutate, defined as follows.

Function hvl-mutate(x).

input: A syntax tree x

Perform one of the following operations uniformly at random;

insert(x):	Choose a terminal $t \in L$ uniformly at random;
	Choose a node $u \in x$ uniformly at random;
	Replace u by a join node whose children are t and u;
substitute(x):	Choose a terminal $t \in L$ uniformly at random;
	Choose a leaf node $u \in x$ uniformly at random;
	Replace u by t;
delete(x):	Choose a leaf node $u \in x$ randomly with parent p and sibling v;
	Replace p by v and delete p and u;

The particular GP problem that we consider here is the ORDER problem introduced in [15]. In ORDER, the only function symbol is a *join node* (denoted by J), which is binary. The terminal set is a set of $2n$ variables $\{v_i, \bar{v}_i \mid i \in \{1, \ldots, n\}\}$, where $\bar{v}_i$ is considered the complement of v_i. Hence, $F := \{J\}$ and $L := \{v_1, \bar{v}_1, v_2, \bar{v}_2, \ldots, v_n, \bar{v}_n\}$.

For a given solution x, the fitness value is computed by parsing the represented tree via an in-order traversal. A variable v_i contributes a value of 1 to the fitness of x iff v_i is visited in the in-order traversal before every $\bar{v}_i$ in the tree. In this case, we call v_i expressed and unexpressed otherwise. A variable v_i is called *redundant* if it occurs more than once in the tree; in this case the variable contributes only once to the fitness value and all duplicates can be removed without decreasing the ORDER value. The same holds for all negated variables $\bar{v}_i$, $1 \leq i \leq n$, and negated variables are always redundant. The goal of the ORDER problem is to maximize the fitness value ORDER. For a given syntax tree x the computation of ORDER(x) is as follows.

Function ORDER(x).

input: A syntax tree x

$l \leftarrow$ an empty leaf list;
$S \leftarrow$ an empty statement list;
Parse x with an in-order traversal and insert each leaf at the rear of l;
Generate S by parsing l front to rear and adding ("expressing") a leaf to S only if it or its complement are not yet in S (i.e., have not yet been expressed);
return $|\{v_i \in S\}|$;

For example, let x be a syntax tree that yields the in-order parse $l = (v_1, \bar{v}_4, v_2, \bar{v}_1, v_3, \bar{v}_6)$. Then $S = (v_1, \bar{v}_4, v_2, v_3, \bar{v}_6)$ and ORDER(x) $= 3$ since $v_1, v_2, v_3 \in S$.

The multi-objective problem MO-ORDER takes the complexity C of a syntax tree as the second objective function. The bi-objective problem MO-ORDER(x) is defined as

$$\text{MO-ORDER}(x) = \big(\text{ORDER}(x), C(x)\big),$$

where $C(x)$ denotes the number of leaves in the syntax tree x. It imposes a trade-off between the ORDER-value and the complexity of a syntax tree. Note that in this case, the goal is to maximize the ORDER-value, but minimize the complexity. In this case a Pareto optimal solution corresponds to a tree of minimal complexity for a particular ORDER-value. This can be contrasted to the pseudo-Boolean problems in Sections 3 and 4 where both functions were maximized.

Let $T_{init,min}$ be the size of the smallest tree in the initial population. In the following, we show that $(\mu + 1)$ SIBEA computes an optimal population efficiently if the population can cover the whole Pareto front.

Theorem 8. *If $\mu \geq n + 1$, the expected optimization time of the $(\mu + 1)$ SIBEA on MO-ORDER is $O(\mu T_{init,min} + \mu n \log n)$.*

Proof. Any population can manifest at most $n + 1$ distinct fitness values, which implies that the number of non-dominated solutions is upper bounded by $n + 1$. This means that if a Pareto optimal solution is found, it will remain in the population during the rest of the algorithm run. The expected time until an empty tree is included in the population is $O(\mu T_{init,min})$. Since the maximum Pareto front size is $n + 1$, we need to generate at most n other Pareto optimal solutions. Let P be a population containing a subset of Pareto optimal solutions with ORDER-value j for every $0 \leq j \leq i$ for some $0 < i < n$. Furthermore, suppose P does *not* contain a Pareto optimal solution with ORDER-value $i + 1$. It is possible for mutation to produce from P an intermediate population P' that is equal to P but also contains a new Pareto optimal individual with ORDER-value $i + 1$. This occurs if HVL-prime mutation inserts one of the $n - i$ unexpressed variables into a Pareto optimal solution with ORDER-value i. The probability of such a step can be calculated as follows.

- The probability of choosing the correct solution (i.e., one with ORDER-value i) from the population is at least $1/\mu$.
- The probability that HVL-prime performs an insertion mutation is $1/3$.
- The probability of choosing an unexpressed variable for insertion is $(n - i)/2n$.

The expected time until all Pareto optimal solutions are produced after having included the empty tree in the population is therefore bounded above by

$$\sum_{i=0}^{n-1} \left(\frac{n-i}{6\mu n}\right)^{-1} = 6\mu n \cdot \sum_{j=1}^{n} \frac{1}{j} = O(\mu n \log n).$$

Adding the time for both phases yields the claimed bound. □

Note that for ORDER, a Pareto optimal solution consists of a syntax tree that contains no redundant variables. Thus, a Pareto optimal solution x with ORDER(x) $= i$ has complexity $C(x) = i$. It follows that all points on the Pareto front lie on the unique line that contains the objective vectors $(0, 0)$ and (n, n). Assuming $\mu - 1$ divides n, this implies that a population with optimal hypervolume must have all μ points equally distributed on the Pareto front.

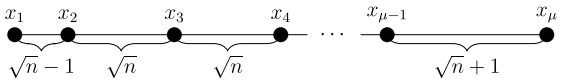

Fig. 5. The intervals corresponding to population P^w for ORDER.

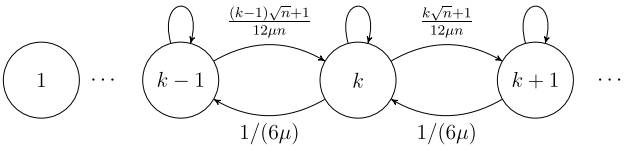

Fig. 6. Markov chain for interval dynamics for $(\mu+1)$ SIBEA on ORDER.

The following theorem will show that, with probability going to one exponentially fast, $(\mu+1)$ SIBEA on MO-ORDER requires exponential time to reach the optimal population state starting with some specific initial populations. We use similar arguments as for the lower bound given in Theorem 3 for OneMinMax. We again index the population by the order of their ORDER-value, that is, $\text{ORDER}(x_{i-1}) \leq \text{ORDER}(x_i)$ for all $i \in \{2,3,\ldots,n\}$ and denote $\iota(x_i)$ as the interval between the ORDER-value of x_i and x_{i-1}. We define a pathological initial population P^w in which (1) $\mu = \sqrt{n}+1$, (2) the extreme points already exist in P^w, and (3) the intervals are as follows.

$$\iota(x_i) = \begin{cases} \sqrt{n}-1 & \text{if } i = 2; \\ \sqrt{n}+1 & \text{if } i = \mu; \\ \sqrt{n} & \text{if } i \in \{3,\ldots,\mu-1\}. \end{cases}$$

Note that this is similar to the initial population used in the proof of Theorem 3, except the large interval now lies to the right and the small interval to the left (see Fig. 5).

Theorem 9. *If $\mu = \sqrt{n}+1$, and the algorithm starts with $P^{(1)} = P^w$, the expected optimization time of $(\mu+1)$ SIBEA on MO-ORDER is exponential with probability $1-2^{-\Omega(n)}$.*

Proof. By Theorem 2, the population $P^{(t)}$ is optimal when $\iota(x) = \sqrt{n}$ for all $x \in P^{(t)} \setminus \{x_1\}$. Starting at P^w, in order to reach the solution with the optimal hypervolume, the small interval $\iota(x_i) = \sqrt{n}-1$ and the large interval $\iota(x_{i+1}) = \sqrt{n}+1$ must meet for some $i \in \{2,\ldots,\mu\}$. Only from this state a single mutation is possible to reach the optimum.

Let T denote the time step in which $P^{(T)}$ is optimal with respect to the hypervolume indicator. We consider the position of the large and small intervals during all times $t < T$. Let $a = a(t)$ be the index at time t such that $\iota(x_a) = \sqrt{n}-1$ and let $b = b(t)$ be the index at time t such that $\iota(x_{b+1}) = \sqrt{n}+1$. The optimal solution is only reachable from the state in which $a = b$. Before this occurs, at least one of the following conditions must hold at some time $t < T$. (1) $\text{ORDER}(x_a) \geq \frac{3n}{4}$, or (2) $\text{ORDER}(x_b) \leq \frac{3n}{4}$.

We now prove that either of these conditions occur only after exponential time with high probability. For the second condition, note that probability of a mutation that moves the large interval to the left is $(n - \text{ORDER}(x_b))/(12\mu n)$. Here, $1/\mu$ is the probability of choosing the correct individual x_b from the population, $(n - \text{ORDER}(x_b))/6n$ is the probability of inserting an unexpressed variable and $1/2$ is the probability of keeping the newly generated offspring.

Meanwhile, the probability of moving the large interval to the right (if possible) is $\frac{1}{6\mu}$ where $1/\mu$ is the probability of choosing the correct solution x_{b+1} from the population, $1/3$ is the probability of choosing deletion in HVL-prime (which deletes some expressed variable and reduces the ORDER value) and $1/2$ is the probability of keeping the newly generated offspring.

Let X_t be the set of variables describing a Markov process over a finite state space $S = \{1,2,\ldots,(\lceil n/4\rceil - 1)/\sqrt{n}\} \cup F$ where the k-th state corresponds to the event in which $n - \text{ORDER}(x_{b(t)+1}) = k\sqrt{n}+1$, that is, the large interval has moved into the k-th position starting from the right. The chain is absorbed into a final accepting state F if the small interval meets it, which we later show also takes exponential time. Letting T denote the time when the intervals first meet, for all $0 \leq t < T$, $X_{t+1} - X_t \in \{-1,0,1\}$. Using the above transition probabilities (illustrated in Fig. 6), for all $0 < t < T$ and $1 \leq X_t \leq (\lceil n/4\rceil - 1)/\sqrt{n}$,

$$\Pr(X_{t+1} - X_t = -1 \mid X_t) \geq \delta \cdot \Pr(X_{t+1} - X_t = 1 \mid X_t),$$

where

$$\delta \geq \frac{1/(6\mu)}{(n - \text{ORDER}(x_b))/(12\mu n)} > (1/6\mu)/(1/48\mu) = 8.$$

The final inequality comes from $\text{ORDER}(x_b) > 3n/4$. By the Local Gambler's Ruin Theorem [16], the time t that $b(t)$ becomes small enough so that $\text{ORDER}(x_{b(t)}) \leq 3n/4$ conditioned on the event that it has not yet met the small interval is at least $\delta^{1/3\cdot(1-1/(4\sqrt{n}))\cdot n} > 8^{\Omega(n)}$ with probability $1-2^{-\Omega(n)}$.

A symmetric argument shows that with probability $1 - 2^{-\Omega(n)}$ it takes exponential time for a to reach a position in which $\text{ORDER}(x_a) \geq 3n/4$. Satisfying either condition thus takes at least exponential time with high probability, and the claim follows. □

6. Conclusions

We have contributed to the theoretical understanding of hypervolume-based evolutionary algorithms and have developed rigorous results on the runtime of these algorithms. Our results specifically show that LOTZ is easy to optimize for a wide range of μ. Furthermore, we have shown that even simple problems such as OneMinMax can lead to plateaus that are hard to leave for $(\mu+1)$ SIBEA. This implies that the optimization time of this algorithm becomes exponential when starting with such a worst-case initial population. Our analyses for multi-objective genetic programming show that hypervolume-based algorithms solve the ORDER problem efficiently if the population is able to cover the whole Pareto front. For a smaller population size we have pointed out that there are situations leading to an exponential optimization time.

References

[1] Anne Auger, Johannes Bader, Dimo Brockhoff, Eckart Zitzler, Investigating and exploiting the bias of the weighted hypervolume to articulate user preferences, in: Genetic and Evolutionary Computation Conference (GECCO), 2009, pp. 563–570.
[2] Anne Auger, Johannes Bader, Dimo Brockhoff, Eckart Zitzler, Hypervolume-based multiobjective optimization: theoretical foundations and practical implications, Theoret. Comput. Sci. 425 (2012) 75–103.
[3] Rudolf Berghammer, Tobias Friedrich, Frank Neumann, Convergence of set-based multi-objective optimization, indicators and deteriorative cycles, Theoret. Comput. Sci. 456 (2012) 2–17.
[4] Nicola Beume, Boris Naujoks, Michael T.M. Emmerich, SMS-EMOA: multiobjective selection based on dominated hypervolume, European J. Oper. Res. 181 (2007) 1653–1669.
[5] Karl Bringmann, Tobias Friedrich, Approximating the volume of unions and intersections of high-dimensional geometric objects, Comput. Geom. Theory Appl. 43 (2010) 601–610.
[6] Karl Bringmann, Tobias Friedrich, Approximation quality of the hypervolume indicator, Artificial Intelligence 195 (2013) 265–290.
[7] Dimo Brockhoff, Optimal μ-distributions for the hypervolume indicator for problems with linear bi-objective fronts: exact and exhaustive results, in: Eighth International Conference on Simulated Evolution and Learning (SEAL '10), 2010, pp. 24–34.
[8] Dimo Brockhoff, Tobias Friedrich, Frank Neumann, Analyzing hypervolume indicator based algorithms, in: Tenth International Conference on Parallel Problem Solving from Nature (PPSN X), 2008, pp. 651–660.
[9] C.A. Coello Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, New York, 2002.
[10] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms, Wiley, Chichester, UK, 2001.
[11] Greg Durrett, Frank Neumann, Una-May O'Reilly, Computational complexity analysis of simple genetic programing on two problems modeling isolated program semantics, in: Eleventh International Workshop on Foundations of Genetic Algorithms (FOGA '11), 2011, pp. 69–80.
[12] Michael T.M. Emmerich, Nicola Beume, Boris Naujoks, An EMO algorithm using the hypervolume measure as selection criterion, in: Third International Conference on Evolutionary Multi-Criterion Optimization (EMO '05), 2005, pp. 62–76.
[13] Datamodeler 8.06, Evolved Analytics LLC, 2012.
[14] Oliver Giel, Per Kristian Lehre, On the effect of populations in evolutionary multi-objective optimisation, Evol. Comput. 18 (3) (2010) 335–356.
[15] David E. Goldberg, Una-May O'Reilly, Where does the good stuff go, and why? How contextual semantics influences program structure in simple genetic programming, in: European Conference on Genetic Programming (EuroGP), 1998, pp. 16–36.
[16] Edda Happ, Daniel Johannsen, Christian Klein, Frank Neumann, Rigorous analyses of fitness-proportional selection for optimizing linear functions, in: Genetic and Evolutionary Computation Conference (GECCO), 2008, pp. 953–960.
[17] John R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.
[18] Marco Laumanns, Lothar Thiele, Eckart Zitzler, Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions, IEEE Trans. Evol. Comput. 8 (2) (2004) 170–182.
[19] Frank Neumann, Computational complexity analysis of multi-objective genetic programming, in: Genetic and Evolutionary Computation Conference (GECCO), 2012, pp. 799–806.
[20] Anh Quang Nguyen, Andrew M. Sutton, Frank Neumann, Population size matters: rigorous runtime results for maximizing the hypervolume indicator, in: Genetic and Evolutionary Computation Conference (GECCO), 2013, pp. 1613–1620.
[21] David Williams, Probability with Martingales, Cambridge University Press, Cambridge Mathematical Textbooks, 1991.
[22] Eckart Zitzler, Simon Künzli, Indicator-based selection in multiobjective search, in: Eighth International Conference Parallel Problem Solving from Nature (PPSN VIII), 2004, pp. 832–842.