Running time analysis of a multi-objective evolutionary algorithm on a simple discrete optimization problem

Marco Laumanns Lothar Thiele Eckart Zitzler
Emo Welzl Kalyanmoy Deb

TIK-Report No. 123
Institut für Technische Informatik und Kommunikationsnetze, ETH Zürich
Gloriastrasse 35, ETH-Zentrum, CH–8092 Zürich, Switzerland

Abstract

For the first time, a running time analysis of a multi-objective evolutionary algorithm for a discrete optimization problem is given. To this end, a simple pseudo-Boolean problem (LOTZ: leading ones - trailing zeroes) is defined and a population-based optimization algorithm (FEMO). We show, that the algorithm performs a black box optimization in $\Theta(n^2 \log n)$ function evaluations where n is the number of binary decision variables.

1 Introduction

Evolutionary Algorithms (EAs) are probabilistic search heuristics that mimic principles of natural evolution. They are often used to solve optimization problems, and recently also for problems with multiple objectives, for an overview see e.g. (Deb 2001). In multi-objective optimization, the aim is to find or to approximate the set of Pareto-optimal (or non-dominated) solutions.

Theoretic work on convergence in evolutionary multi-objective optimization is mainly due to Rudolph (1998a, 1998b, 2001), Rudolph and Agapie (2000), Hanne (1999, 2001), Van Veldhuizen (1999), and Laumanns et al. (2001). These studies exclusively deal with the limit behavior. Under appropriate conditions for the variation and the selection operators, global convergence to the optimum can be guaranteed in the limit.

On the other hand, we are often interested in a quantitative analysis, specifically the expected running time for a given class of problems and the success

probability for a given optimization time. Many results on single-objective evolutionary algorithms are described in Rudolph (1997). For the optimization of pseudo-Boolean functions an extensive theory has been built up by Wegener et al., see e.g. Wegener (2001), Droste, Jansen, and Wegener (1998, 2002), or for a methodological overview Wegener (2000).

To the best of our knowledge, no results on the running time of evolutionary algorithms are available in the multi-objective case. The purpose of this paper is to make a first step into this direction by analyzing different simple multi-objective evolutionary algorithms (MOEAs) on a two-objective model problem. In particular, the following results are described in the paper:

- The well known "Leading Ones" problem is generalized to two dimensions. The new problem class is called LOTZ (Leading Ones - Trailing Zeros).
- A simple evolutionary multi-objective optimization algorithm is defined (SEMO - Simple Evolutionary Multi-objective Optimizer). Its expected running time on the above problem is shown to be $\Theta(n^3)$.
- The algorithm is improved by a fair sampling strategy of the population (FEMO - Fair Evolutionary Multi-objective Optimizer). Its running time on the above problem is $\Theta(n^2 \log n)$ with a high probability of $1 - O(1/n)$.

The model problem and its characteristics are introduced in section 2. It is a multi-objective extension of the "Leading Ones" problem which has been thoroughly analyzed for example by Rudolph (1997) and Droste et al. (2002). The algorithms are described and analyzed in sections 3 and 4. They are instances of a steady state $(\mu + 1)$-EA with variable population size and differ in the manner how the parents are sampled from the population.

2 The Model Problem

As the example problem for this analysis, we consider the maximization of a 2-dimensional vector valued function LOTZ which maps n binary decision variables to 2 objective functions.

Definition 1
The pseudo-Boolean function $\text{LOTZ} : \{0,1\}^n \to \mathbb{N}^2$ *is defined as*

$$\text{LOTZ}(x_1, \ldots, x_n) = \left(\sum_{i=1}^{n} \prod_{j=1}^{i} x_j, \sum_{i=1}^{n} \prod_{j=i}^{n} 1 - x_j \right)$$

The abbreviation LOTZ stands for "Leading Ones, Trailing Zeroes" and means that we want to simultaneously maximize the number of leading ones and trailing zeroes in a bit-string. The first component, the LEADINGONES function, has been analyzed in detail by (Rudolph 1997) and (Droste, Jansen, and Wegener 2002).

As there is no single search point that maximizes both components simultaneously, we want to find the whole set of non-dominated points based on the concept of Pareto optimality, here defined for binary decision variables.

Definition 2 (Pareto optimality, Pareto set)
Let $f : X \to F$ where $X \subseteq \{0,1\}^n$ and $X \subseteq \mathbb{R}^m$. A decision vector $x^* \in X$ is Pareto optimal *if there is no other $x \in X$ that* dominates *x^*. x dominates x^*, denoted as $x \succ x^*$ if $f_i(x) \geq f_i(x^*)$ for all $i = 1, \ldots, m$ and $f_i(x) > f_i(x^*)$ for at least one index i. The set of all Pareto optimal decision vectors X^* is called* Pareto set. *$F^* = f(X^*)$ is the set of all Pareto optimal objective vectors or the* Pareto front.

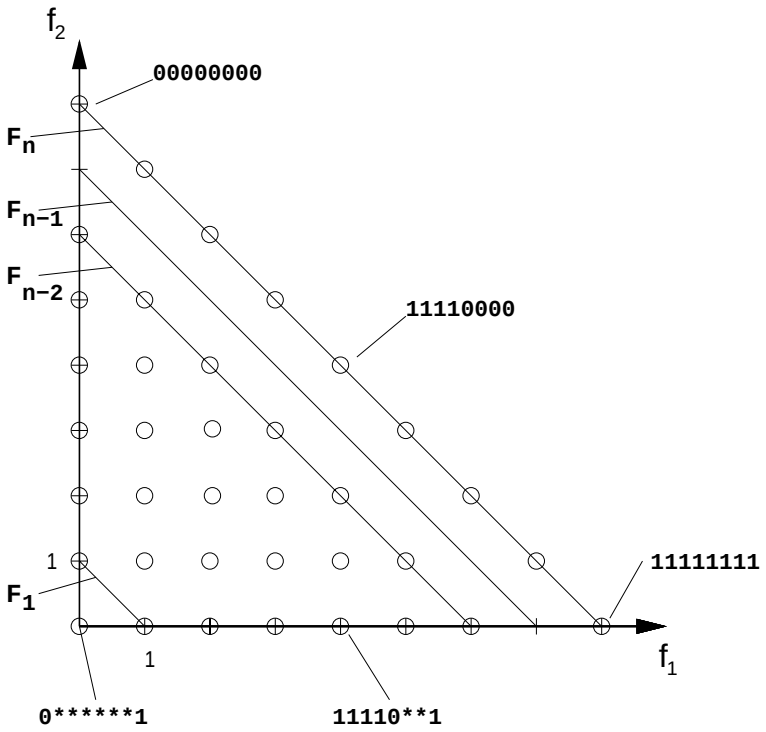

Figure 1: Objective space of the LOTZ function with $n = 8$.

The LOTZ function has some interesting properties, see also Figure 1. The decision space X contains 2^n different elements, which are mapped to only $1/2n^2 + 3/2n + 1 = O(n^2)$ different objective vectors.

The objective space can be partitioned into $n+1$ sets $F_i, i = 0, \ldots, n$, where the index i corresponds to the sum of both objective values, i.e. $(f_1, f_2) \in F_i$ if $i = f_1 + f_2$. Obviously, F_n represents the Pareto front F^*. The subdomains X_i are defined as the sets containing all decision vectors which are mapped to elements of F_i. They are of the form $1^a 0 *^{(n-i)} 10^b$ with $a+b = i$.

The cardinality of the Pareto set $X^* = X_n$ is $|X_n| = n+1$ and we also have $n+1$ Pareto optimal objective vectors as $|F_n| = n+1$. The next set F_{n-1} is empty. For the remaining sets with $i = 0, \ldots, n-2$ we have $|F_i| = i+1$ and $|X_i| = |F_i| \cdot 2^{n-2-i}$.

2.1 How difficult is it to find the whole Pareto set?

How long does it take to optimize the LOTZ function? Droste et al. (2002) have proved that the expected running time of a (1+1) EA on LEADINGONES is $\Theta(n^2)$. Using the same algorithm with an appropriate generalization of the acceptance criterion (either accepting only dominating offspring or by using a weighted sum as a scalar surrogate objective) will certainly lead to finding *one* element of the Pareto set in the same amount of time.

To find all different Pareto optimal points with such a (1+1) EA we can consider the multi-start option, i.e. to run to the EA several times, and collect all non-dominated solutions in an archive. For the acceptance criterion based on the dominance relation, the random variable describing the number of ones in the final solution of each single run follows a binomial distribution with $p = 0.5$. Hence the probability of finding the "outer" points of the Pareto set decreases exponentially. This would mean that the running time of this strategy until all Pareto optimal points are found is exponentially large in n.

Another possibility would be to use the multi-start option together with a weighted sum of the objective values. However, an appropriate choice of the weights is very difficult. In our case, equal weights would lead to the same situation as before, with a very low probability to reach the outer points. Any other selection of weights will let the sequence of search points converge to one of the outer points of the Pareto set. The remaining points must be found "on the way", but the probability of such events is not easy to calculate. Even if we could supply $n+1$ different weights corresponding to each of the $n+1$ Pareto optimal points, this strategy would still need $\Theta(n^3)$ steps. But as the exact shape of the Pareto set is usually not known in advance, this strategy is not feasible in practice.

A last possibility would be to use a simple strategy known from classical multi-objective function optimization. In this case, we optimize only one objective, e.g. the number of leading ones, and constrain the other objective

to be strictly larger than its value obtained in the previous optimization run. Therefore, we find all n Pareto vectors in n runs of a single-objective EA with an additional constraint. At the best, this strategy again needs $\Theta(n^3)$ steps.

The above discussion indicates that a (1+1) strategy may not be the best approach for solving multi-objective optimization problems. Moreover, most of the current multi-objective optimization algorithms use the concept of an archive that maintains a set of Pareto optimal vectors of all decision vectors visited so far. This indicates that the concept of a population is vital in multi-objective evolutionary optimization. In the next sections we propose two simple population-based steady state EAs which can solve the example problem efficiently.

3 A Simple Evolutionary Multi-Objective Optimizer (SEMO)

At first, we analyze a simple population-based multi-objective EA. This algorithm comprises a population of variable size which stores all non-dominated individuals. From this population a parent is drawn according to some probability distribution and mutated by flipping a randomly chosen bit. For algorithm 1 we consider a uniform distribution for selecting the parent.

Algorithm 1 Simple Evolutionary Multi-Objective Optimizer (SEMO)

1: Choose an initial individual x uniformly from $X = \{0,1\}^n$
2: $P \leftarrow \{x\}$
3: **loop**
4: Select one element x out of P uniformly.
5: Create offspring x' by flipping a randomly chosen bit.
6: $P \leftarrow P \setminus \{z \in P | x' \succ z\}$
7: **if** $\nexists z \in P$ such that $(z \succ x' \vee f(z) = f(x'))$ **then**
8: $P \leftarrow P \cup \{x'\}$
9: **end if**
10: **end loop**

An appropriate archiving strategy (Laumanns et al. 2001) is assumed to prevent the population from growing exponentially. For this study it suffices to ensure that each new accepted solution must have different objective function values (line 7).

3.1 Running Time Analysis of SEMO applied to LOTZ

The running time of an algorithm equals the number of necessary evaluations of the objective function.

For the analysis we divide the run of the SEMO into two distinct phases: the first phase lasts until the first Pareto-optimal individual has entered the population, and the second phase ends when the whole Pareto set has been found.

Lemma 1 (Expected running time for phase 1)
The expected running time of Alg. 1 until the first Pareto-optimal point is found is $O(n^2)$.

Proof. Note that during this first phase the population will consist of one individual only, as a mutation changing the objective values yields either a dominating or a dominated individual. Hence, if an offspring is accepted, it will replace the parent from which it was produced. From any subset X_i only points in $X_j, j > i$ are accepted. As there is always a one-bit mutation leading to next subset, the probability of improvement is at least $1/n$. As there are at most $n-1$ such steps necessary (X_{n-1} is empty) the expected time is at most n^2. □

Lemma 2 (Expected running time for phase 2)
After the first Pareto-optimal point is found, the expected running time of Alg. 1 until all Pareto-optimal points are found is $\Theta(n^3)$

Proof. We partition this phase into $n-1$ different sub-phases. Sub-phase i lasts from the time when $i-1$ Pareto-optimal solutions have been found to the time when the next solution is found. T_i is a random variable denoting the duration of sub-phase i and the random variable T is the sum of these times. As we always have a contiguous subset of the Pareto set, only the individuals corresponding to the outer points of this subset can create a new Pareto-optimal point. The probability $p_s(i)$ to sample such a candidate in phase i is at least $1/i$ and at most $2/i$. A subsequent mutation has a success probability of at least $1/n$ and at most $2/n$. Hence, $ni/4 \leq E(T_i) \leq ni$. As $T = \sum_{i=1}^{n-1} T_i$, $1/8n^3 - 1/8n^2 \leq E(T) \leq 1/2n^3 - 1/2n^2$. □

From the concatenation of the two phases the following Corollary can be derived.

Corollary 1 (Expected running time Alg. 1)
The expected running time of Alg. 1 until all Pareto-optimal points are found is $\Theta(n^3)$

For this problem, the simple population-based SEMO is at least as good as any potential multi-objective adaptation of the (1+1)-EA discussed in the previous section. But is there room for further improvement? As it takes about n^2 steps to find *one* Pareto-optimal point, there is no hope to find the whole Pareto set in less time. But the time to generate all Pareto-optimal points can be reduced substantially.

4 The Fair Evolutionary Multi-Objective Optimizer (FEMO)

The main weakness of the SEMO for the objective function under consideration lies in the fact that a large number of mutations are allocated to parents whose neighborhood had already been explored sufficiently. On the other hand, an optimal sampling algorithm would use always the most promising parent at the border of the current population. Of course, this information is not available in a black box optimization scenario.

The uniform sampling leads to a situation, where the Pareto-optimal individuals have been sampled unevenly depending on when each individual entered the population.

The following *fair* sampling strategy guarantees that the end all individuals receive about the *same* number of samples.

Algorithm 2 Fair Evolutionary Multi-Objective Optimizer (FEMO)

1: Choose an initial individual x uniformly from $X = \{0,1\}^n$
2: $w(x) \leftarrow 0$
3: $P \leftarrow \{x\}$
4: **loop**
5: Select one element x out of $\{y \in P | w(y) \leq w(z) \forall z \in P\}$ uniformly.
6: $w(x) \leftarrow w(x) + 1$
7: Create offspring x' by flipping a randomly chosen bit.
8: $P \leftarrow P \setminus \{z \in P | x' \succ z\}$
9: **if** $\nexists z \in P$ such that $(z \succ x' \vee f(z) = f(x'))$ **then**
10: $P \leftarrow P \cup \{x'\}$
11: **end if**
12: **end loop**

Algorithm 2 implements this strategy by counting the number of offspring each individual produces. The sampling procedure deterministically chooses the individual which has produced the least number of offspring so far, ties are broken randomly.

4.1 Running Time Analysis of FEMO applied to LOTZ

For the analysis of Algorithm 2 we focus only on the second phase as the first phase is identical to the simple Algorithm 1 described before.

Once the first two Pareto-optimal points are found, there is exactly one possible parent for each of the remaining $n-1$ points. We are interested in bounding the number of mutations that must be allocated to each of these $n-1$ parents in order to have at least one successful mutation that leads to the desired child.

Lemma 3 (Minimal success probability)
Let p be the success probability for each single mutation. With probability at least $1-n^{1-c}$ all $n-1$ the remaining offsprings have been constructed in at most $c \cdot 1/p \cdot \log n$ mutation trials for each corresponding parent.

Proof. For each individual, the probability of having at least $t = c \cdot 1/p \cdot \log n$ non-successful mutation is bounded above by

$$(1-p)^t = (1-\frac{1}{1/p})^{c/p \log n} = (1-\frac{1}{1/p})^{1/p^{c \log n}} \leq \left(\frac{1}{e}\right)^{c \log n} = \frac{1}{n^c}$$

There are $n-1$ individuals that must be produced with the given number of trials. These events are independent, so the probability that at least one individual needs more than t trials is bounded above by $\frac{n-1}{n^c} \leq n^{1-c}$. □

Lemma 4 (Maximal success probability)
Let $k \in \{1, \ldots, n\}$ and $a = k/n$. The probability that $k = a \cdot n$ individuals have been produced in $c \cdot 1/p \cdot \log n$ mutation steps each is not greater than $(e^a)^{-n^{(1-c-c/n)}}$.

Proof. The probability that a parent has created a certain offspring within the first $t = c \cdot 1/p \cdot \log n$ trials is $1-(1-p)^t$. The probability that this happens independently for a selection of k such pairs can thus be bounded as

$$(1-(1-p)^t)^k \leq (1-\frac{1}{n^{c\frac{n+1}{n}}})^{an} \leq e^{-\frac{an}{n^{c(n+1)/n}}} = (e^a)^{-n^{(1-c-c/n)}}$$

□

Now we can translate the number of mutations that are needed into the number of samples from the objective function.

Theorem 1 (Running time bounds)
With probability at least $1 - O(1/n)$ *the number of objective function evaluations* T *Algorithm 2 from the discovery of the first two Pareto-optimal points until the whole Pareto set has been found lies in the interval* $[1/4 \cdot 1/p \cdot n \log n, 2 \cdot 1/p \cdot n \log n]$. *Hence,* $Prob\{T = \Theta(1/p \cdot n \log n)\} = 1 - O(1/n)$.

Proof. Let the Pareto-optimal points be indexed according the the order in which they have entered the set P. Let $k \in \{0, \ldots, n\}$ be the index of the individual that required the largest number of mutations to be produced. Because of Lemma 3 this individual k did not need more than $t = 2/p \log n$ trials with probability $1 - O(1/n)$.

What remains to be shown for the upper bound is that no node will be *sampled* more than t times during the algorithm. This can be guaranteed since there is always a candidate $x \in P$ with $w(x) \leq t$ (the element that has most recently been added to P). Hence, any element whose weight has reached t will never be sampled again. As there are n such elements, each of which is sampled at most t times, the total number of samples (steps) the algorithm takes does not exceed $T = n \cdot t = 2 \cdot 1/p \cdot n \log n$.

For the lower bound, Lemma 4 assures that with a probability of $1 - \sqrt{e}^{-n^{(0.5-0.5/n)}}$ there is an individual in the second half which needs at least $1/2 \cdot 1/p \cdot \log n$ trials. Hence, all individuals in the first half have been sampled at least $1/2 \cdot 1/p \cdot \log n - 1$ times each. Of course, all individuals in the second half must be sampled at least once. The summation over all nodes gives a total number of samples of at least $1/4 \cdot 1/p \cdot n \log n$ with probability $1 - O(1/n)$. □

As before, the time to find the first one (or two) elements of the Pareto set can be neglected and the total running time is mainly determined by Theorem 1. For our case the mutation success probability is $p = 1/n$ which leads with a high probability to a running time of $\Theta(n^2 \log n)$. This is a considerable improvement in comparison to any multi-start strategy of a single objective EA and to the SEMO algorithm.

5 Concluding Remarks

The purpose of this report is to initiate further research on running time analysis of multi-objective evolutionary optimization algorithms.

In particular, we have been able to show that a simple multi-objective evolutionary search strategy involving the concept of an archive or population

is able to efficiently solve a discrete optimization problem. The running time $\Theta(n^2 \log n)$ beats that of a single-objective optimizer with a multi-start strategy which is at least $\Theta(n^3)$.

6 Acknowledgments

The research has been funded by the Swiss National Science Foundation (SNF) under the ArOMA project 2100-057156.99/1.

References

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley.

Droste, S., T. Jansen, and I. Wegener (1998). A rigorous complexity analysis of the (1+1) evolutionary algorithm for separable functions with boolean inputs. *Evolutionary Computation 6*(2), 185–196.

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

Hanne, T. (1999). On the convergence of multiobjective evolutionary algorithms. *European Journal Of Operational Research 117*(3), 553–564.

Hanne, T. (2001). Global multiobjective optimization with evolutionary algorithms: Selection mechanisms and mutation control. In E. Zitzler et al. (Eds.), *Evolutionary Multi-criterion Optimization (EMO 2001), Proc.*, Lecture Notes in Computer Science Vol. 1993, Berlin, pp. 197–212. Springer.

Laumanns, M., L. Thiele, K. Deb, and E. Zitzler (2001, May). On the convergence and diversity-preservation properties of multi-objective evolutionary algorithms. Technical Report 108, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland.

Rudolph, G. (1997). *Convergence Properties of Evolutionary Algorithms*. Verlag Dr. Kovač, Hamburg.

Rudolph, G. (1998a). Evolutionary search for minimal elements in partially ordered sets. In *Evolutionary Programming VII – Proc. Seventh Annual Conf. on Evolutionary Programming (EP-98)*, San Diego CA. The MIT Press, Cambridge MA.

Rudolph, G. (1998b). On a multi-objective evolutionary algorithm and its convergence to the pareto set. In *IEEE Int'l Conf. on Evolutionary Computation (ICEC'98)*, Piscataway, pp. 511–516. IEEE Press.

Rudolph, G. (2001). Evolutionary search under partially ordered fitness sets. In *Proceedings of the International Symposium on Information Science Innovations in Engineering of Natural and Artificial Intelligent Systems (ISI 2001)*, pp. 818–822.

Rudolph, G. and A. Agapie (2000). Convergence properties of some multi-objective evolutionary algorithms. In A. Zalzala and R. Eberhart (Eds.), *Congress on Evolutionary Computation (CEC 2000)*, Volume 2, Piscataway, NJ, pp. 1010–1016. IEEE Press.

Van Veldhuizen, D. A. (1999, June). *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. Ph. D. thesis, Graduate School of Engineering of the Air Force Institute of Technology, Air University.

Wegener, I. (2000). Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. Technical Report CI-99/00, SFB 531, Universität Dortmund.

Wegener, I. (2001). Theoretical aspects of evolutionary algorithms. In *ICALP 2001*, Volume 2076 of *LNCS*, pp. 64–78. Springer-Verlag.