Maximizing Population Diversity in Single-Objective Optimization

Tamara Ulrich Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich
8092 Zurich, Switzerland
firstname.lastname@tik.ee.ethz.ch

ABSTRACT

Typically, optimization attempts to find a solution which minimizes the given objective function. But often, it might also be useful to obtain a set of structurally very diverse solutions which all have acceptable objective values. With such a set, a decision maker would be given a choice of solutions to select from. In addition, he can learn about the optimization problem at hand by inspecting the diverse close-to-optimal solutions.

This paper proposes NOAH, an evolutionary algorithm which solves a mixed multi-objective problem: Determine a maximally diverse set of solutions whose objective values are below a provided objective barrier. It does so by iteratively switching between objective value and set-diversity optimization while automatically adapting a constraint on the objective value until it reaches the barrier. Tests on an nk-Landscapes problem and a 3-Sat problem as well as on a more realistic bridge construction problem show that the algorithm is able to produce high quality solutions with a significantly higher structural diversity than standard evolutionary algorithms.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Diversity in Decision Space

GECCO'11, July 12–16, 2011, Dublin, Ireland.

1. INTRODUCTION

Often, optimization of complex systems is not only concerned with finding a single solution which minimizes an implicitly or explicitly defined objective function. Rather, one is interested in a set of solutions that explore different options while still being of an acceptable quality with respect to the objective function.

Consider the case that an engineer wants to design the electronic system in a car. He is given a fixed cap on the cost which he must satisfy. There are a few standard designs which the engineer could use, e.g. a centralized system where each subsystem is controlled by a central processor, or a distributed design where each subsystem has its own processor. Nevertheless, the engineer would like to know whether there are any other, possibly non-standard designs that satisfy the cost cap, such that he can then select the design which can best be integrated into the given car family. To this end, an algorithm is required that returns a set of designs (i.e. solutions) which are structurally as diverse as possible, but still satisfy the cost cap.

In general, there may be several reasons for an optimization scenario where not a single best solution is of interest but a set of diverse high-quality solutions. At first, the result of the optimization may be only a single step in a complex design process, as in the engineering example above. Due to unknowns in the whole decision process, one would rather be interested in various possible options that explore the solution space and can be evaluated further (maybe based on additional criteria). Secondly, a set of diverse (almost) optimal solutions as the result of an optimization may be used to learn more about the system to be optimized. Finally, optimizations are usually based on a suitable abstraction of the problem, for example in form of an analytic model or a simulation. These models typically contain simplifications and need appropriate parameterizations. This modeling process introduces uncertainties in the objective function. Other reasons for such uncertainties are unknown or time-varying system parameters. An optimization process which yields a single solution may not be sufficient in this case as it reflects only a single possible problem instance. Rather, one would be interested in a diverse set of solutions that provide appropriate decision support.

The above informal problem definition can be interpreted as a special kind of multiobjective optimization, denoted as *mixed multiobjective problem*, where the first goal is to generate solutions which optimize some objective function, and the second goal is to have a final set of solutions which is as diverse as possible with respect to some diversity measure.

In contrast to typical multiobjective problems, where a vector of objective functions is associated to each individual solution, *mixed multiobjective problems* have a different structure: One objective can be described by a function which maps individual solutions to objective values whereas the other objective is defined by a set indicator which maps sets of solutions to objective values. The present paper explores models and methods for this kind of mixed multiobjective optimization.

There exists a large body of methods which integrate diversity preservation into evolutionary search methods, see for example [9, 20]. Most of these methods try to maintain diverse solutions in order to fight the problem of premature convergence during the optimization. To our best knowledge, none of the existing methods explicitly tries to generate a diverse set of solutions as described above. Moreover, known approaches do not directly optimize diversity as a set measure, but rather have some implicit diversity preservation, e.g. through the maintenance of different niches, see also [9]. Section 1.2 will provide a more detailed overview about comparable approaches, including methods that determine solutions which are robust towards uncertainties in the objective function or solutions that reflect sets of local minima.

The following new results are described in the paper:

- An evolutionary algorithm (NOAH) is proposed to solve the *mixed multiobjective optimization problem*, i.e. it determines a set of solutions which (a) have objective values below a provided barrier value and (b) maximize a set diversity measure.
- An algorithm is described that, given a set of solutions, selects a subset of these solutions such that the chosen diversity measure (Solow-Polasky) of the subset is maximum. The algorithm has a low computational complexity such that NOAH has an acceptable run-time behavior.
- An extensive experimental investigation shows the effectiveness of the new approach compared to other evolutionary algorithms.

The paper is structured as follows: Section 1.1 provides a formal problem statement and introduces a simple example to illustrate the problem. Section 1.2 describes relevant related work, whereas Section 2 proposes NOAH, an algorithm to solve the given problem. Section 3 introduces the chosen diversity measure and describes a computationally efficient update procedure. Finally, in Section 4 we experimentally compare our algorithm to other evolutionary algorithms and apply it to a more realistic bridge construction problem.

1.1 Problem Statement and Intuition

We are considering the minimization of a single objective function $f : \mathcal{X} \to \mathbf{R}$. Here, $\mathcal{X}$ denotes the feasible set of solutions in the decision space, i.e. the set of alternatives of the decision problem. A single alternative $x \in \mathcal{X}$ will be denoted as a solution x. The image of $\mathcal{X}$ under f is denoted as the feasible set in the objective space $\mathcal{Z} = f(\mathcal{X}) = \{y \in \mathbf{R} \mid \exists x \in \mathcal{X} : y = f(x)\}$. Therefore, the objective value of a single solution x is $f(x)$.

There are no assumptions about the structure of the decision space, except that a symmetric distance measure $d : \mathcal{X}^2 \to \mathbf{R}$ between two solutions is required. Based on the distance measure we define a diversity measure $D : \mathcal{P}(\mathcal{X}) \to \mathbf{R}$. It is defined on the powerset of the decision space, i.e. all possible subsets of the decision space, and determines for a subset of the decision space its corresponding diversity. Finally, a provided barrier value v is used to determine a constraint on the objective values.

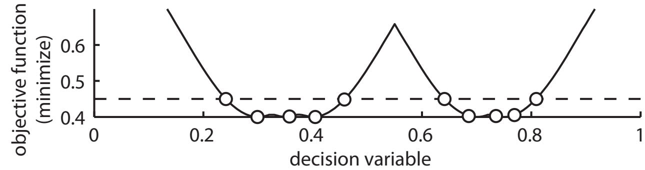

Figure 1: Simple objective function with decision space on x-axis and objective space on y-axis. Circles show solutions found by the NOAH algorithm with barrier 0.5 (dashed horizontal line).

The *mixed multiobjective optimization problem* we are trying to solve can therefore be stated as follows:

Determine a population $P \subseteq \mathcal{X}$ with a given size $|P| = n$ which maximizes the diversity measure D while satisfying the provided barrier v on the objective values:

$$\max_{P \subseteq \mathcal{X}_v, |P|=n} D(P) \text{ where } \mathcal{X}_v = \{x \in \mathcal{X} \mid f(x) \leq v\} \quad (1)$$

In other words, we are trying to find a population P which only contains solutions which are better or equal than the barrier v and which maximizes the diversity measure D. Note that this is *not* the same as multi-modal optimization, where multiple local optima are seeked without considering their quality, see e.g [17]. Neither are we looking for robust solutions or solutions insensitive to change as for example in dynamic environments [3]. Also, we do not consider diversity as an additional independent objective, as we are not interested in diverse but low-quality solutions. Instead, we want diverse solutions that satisfy a certain quality bound.

Let us now present a very simple example. Consider a minimization problem with a one-dimensional real-valued decision space. The objective function is depicted in Figure 1. We would like to find a maximally distributed set of solutions below a given barrier value (horizontal line). Figure 1 shows the case that the decision maker finds all solutions which have an objective value of 0.45 or lower to be acceptable. One possible set of solutions that satisfy the quality constraint and that are well distributed in decision space are shown as circles in the Figure[1].

1.2 Related Work

As stated in the introduction, one of the main reasons why a diverse, close-to-optimal set of solutions is beneficial is that there are uncertainties in the design process and in the modeling of a system. The handling of uncertainties during optimization has been treated before, for an overview see e.g. [12]. Four different categories of uncertainties are distin-

[1]These solutions have been generated using the NOAH algorithm as defined in Algorithm 1 for 450 function evaluations, with the following parameters: $n = 8$, $v = 0.5$, $g = 10$, $r = 4$, $c = 20$ (see Section 2 for more details), using Euclidean distance as a distance measure and using the Solow-Polasky measure as defined in Section 3 for calculating the diversity D.

guished: (1) Subsequent evaluations of the same individual yield different objective values. (2) There are uncertainties in the decision variables. Both categories are usually treated by repeatedly evaluating a single individual in order to get an estimation of its fitness. Further categories are: (3) Uncertainties introduced by the usage of a simplified model of a real-world problem. (4) Objective functions that change over time. Methods dealing with dynamically changing objective functions usually try to introduce or maintain a certain degree of diversity, which will be discussed next.

There are many algorithms which attempt to preserve diversity during an optimization run. The motivation for these methods usually comes from optimizing multimodal problems, where evolutionary algorithms can get stuck in local optima due to genetic drift, see e.g. [9, 20]. One method is to run several populations in parallel with the goal that they will explore different regions in the search space. Island model EAs and parallel EAs fall into this category as well. Usually, there is some exchange between the different populations in the runs, and the main difference between existing algorithms is on how often individuals are exchanged and which individuals are exchanged [9]. Other algorithms run several EAs in sequence, and pass information from one run to the next in order to prevent the following runs to find the same local optimum, see e.g. [2].

Other approaches are based on speciation, an observation from nature which states that first, only individuals from the same species can mate to produce offspring and second, there is a certain amount of geographic separation between individuals from the same species, and only neighboring individuals are eligible for mating. Examples for corresponding algorithmic techniques are assigning individuals to species prior to any selection step and to restrict competition, see [16], or placing mating restrictions on the individuals by assigning a geographic location of each individual, see e.g. [9], or by only allowing individuals within a certain distance of each other to mate [7]. Other methods use fitness sharing, see [10], such that individuals which have a lot of close neighbors have a reduced fitness. Another approach is to use crowding, see [5], where individuals can only be replaced by neighboring individuals.

Many of these algorithms do not optimize diversity explicitly by means of a set measure. Maintaining diversity is used to increase the probability to find the global optimum, or at least different local optima. In this paper, we determine a set of maximally diverse solutions, with the constraint that the solutions must have a certain quality with respect to the given objective function. Note that of course to be able to appropriately set that quality constraint, any standard single-objective optimizer can be used prior to the diversity optimization to calculate the best achievable objective value.

2. NOAH ALGORITHM

In this paper, we propose a new algorithm called NOAH to solve the mixed multiobjective problem. Remember from Section 1.1 that we assume that there is a certain objective value, called the barrier, below which all solutions are acceptable. This barrier value can be flexibly chosen. The algorithm we propose in this paper then generates a population which only contains solutions that are better or equal than this barrier and that are as diverse as possible, see also (1). In case the barrier is set to a value lower than any value the algorithm is able to achieve, then NOAH performs a conventional single objective optimization where solutions with a better objective function value are always more desirable than those with a worse value.

The NOAH algorithm uses two key concepts to solve the above defined *mixed multiobjective optimization* problem: bound adaptation and diversity optimization. Its main structure is shown in Algorithm 1. Each iteration consists of three steps, namely the optimization of the objective function f by means of OBJOPT, the bound adaptation using BOUNDCHANGE and the diversity optimization of $\max D(P)$ in DIVOPT. The iteration stops if all solutions p in the population P have objective value $f(p) \leq v$ or some other termination criterion is satisfied.

Algorithm 1 Mixed multiobjective optimization algorithm NOAH. Input parameters: population size n; barrier value v; minimization of objective function is done for g generations; r solutions remain in the population after bound adaptation; the population diversity converged if it did not improve for a total of c generations.

```
function NOAH(n, v, g, r, c)
    Initialize population P randomly with n solutions
    b = ∞
    while (b > v)∧(termination criterion not reached) do
        P := OBJOPT(P, g, b)
        (P, b) := BOUNDCHANGE(P, b, r)
        P := DIVOPT(P, n, b, c)
    end while
    return P
end function
```

The rationale behind NOAH will be described in some more detail. As mentioned above, in each loop a standard evolutionary algorithm operates for g generations, then the bound is adapted and finally diversity is optimized until it converges. *In other words, objective value and population diversity are jointly optimized by transforming the mixed multiobjective problem into a constrained set diversity optimization.* The constraint is the bound b on the objective values which is adaptively reduced until it reaches the provided barrier value v. The diversity optimization DIVOPT results in a population which is optimized with respect to its diversity $D(P)$ but respects the constraint imposed by the bound b.

Subalgorithms OBJOPT and BOUNDCHANGE are responsible for optimizing the population with respect to the objective function f. OBJOPT receives a population P with n elements and objective values $f(p) \leq b$ and uses a standard evolutionary algorithm for g generations to optimize it. Any optimization algorithm can be used as long as the solutions in the resulting population also have objective values $f(p) \leq b$.

In order to balance diversity optimization and objective value optimization, a bound value b is monotonically decreased during the run in BOUNDCHANGE. The new bound value is set in such a way that at least r individuals in the population are still on or below the new bound. These individuals form the new population.

Finally, DIVOPT maximizes the diversity $D(P)$ under the constraint that the resulting population has again n elements whose objective values are at or below b, i.e. $f(p) \leq b$. The iterative optimization in DIVOPT terminates if the diversity did not improve for a total of c generations. As a result we can state that in each iteration OBJOPT optimizes the population for g generations with respect to the objective function f, then BOUNDCHANGE adaptively adjusts the objective value bound b such that r solutions are on or below the new bound b, and DIVOPT maximizes the diversity while maintaining the bound b. Now, some more details about the different aspects of NOAH are provided.

The objective value optimization OBJOPT uses a simple $\{\mu + \lambda\}$ evolutionary algorithm with $\mu = \lambda = n$ which respects the bound b, see Algorithm 2. The variation function VARIATEPOP may use any appropriate combination of mutation and crossover operators in order to generate a resulting population with n solutions. Its only difference to a standard variation of a given population is that it returns only solutions that have an objective value not worse than b. For example, the internal elementary operators are called as many times as necessary to generate enough feasible individuals. Selection function SELECTOBJ selects a population of n solutions according to some (possibly standard) selection criterion that ensures selection pressure. Note that any other refined strategy can be used for OBJOPT as long as the bound b is respected in the resulting population.

Algorithm 2 Objective value optimization OBJOPT. Input parameters: population P; number of generations g; bound b.

```
function objOpt(P, g, b)
    n := |P|
    for g iterations do
        P' := variatePop(P, b, n)
        P := selectObj({P ∪ P'}, n)
    end for
    return P
end function
```

The strategy to adaptively change the bound value b is described in Algorithm 3. In BOUNDCHANGE, the new bound is set to the minimal value such that at least r solutions are still on or below it. The resulting subset of the population contains all elements with objective values equal or below this new bound.

Algorithm 3 Adaptive change of bound BOUNDCHANGE. Input parameters: population P; current bound b; minimal number of solutions in resulting population r.

```
function boundChange(P, b, r)
    b := minimal x s.t. |{p|p ∈ P, f(p) ≤ x}| ≥ r
    P' := {p ∈ P|f(p) ≤ b}
    return (P, b)
end function
```

The optimization of diversity DIVOPT is described in Algorithm 4. At first, the already described variation operator VARIATEPOP is called which generates a population P' by any appropriate combination of mutation and crossover operators. Again, it returns only solutions that have an objective value not worse than b. The number of generated solutions is chosen such that $\{P \cup P'\}$ has $2n$ solutions (remember that we have chosen $\mu = \lambda = n$). In the selection phase the solutions are selected according to their diversity contribution using the operator SELECTDIV which will be described in much more detail in Section 3. This is in contrast to the standard evolutionary algorithm shown in Algorithm 2, where solutions are selected according to their objective values. Moreover, the diversity optimization is run until there have been c generations in total without an increase in diversity. Note that as soon as the adaptive bound b has reached the user-specified barrier value v, diversity is optimized one more time until it converges and the algorithm NOAH is stopped.

Algorithm 4 Diversity optimization DIVOPT. Input parameters: population P; population size n; bound value b; the total number of generations the diversity did not change for convergence c.

```
function divOpt(P, n, b, c)
    i := 0
    while i < c do
        P' := variatePop(P, b, 2n − |P|)
        P'' := selectDiv({P ∪ P'}, n)
        if D(P'') > D(P) then
            P := P''
        else
            i := i + 1
        end if
    end while
    return P
end function
```

3. DIVERSITY OPTIMIZATION

3.1 Diversity Measure

For the optimization of the diversity in Eq. (1), an appropriate diversity measure $D : \mathcal{P}(\mathcal{X}) \to \mathbf{R}$ has to be selected. In [19] desirable properties of diversity measures are discussed and several measures are compared with respect to these properties. To begin with we do not want to restrict the class of decision spaces that can be considered in NOAH, for example we do not assume that the solutions are given in Euclidean Space. Therefore, we assume only the existence of a symmetric distance measure $d : \mathcal{X}^2 \to \mathbf{R}$ between two solutions. Following [19] we require the following additional conditions:

Monotonicity in Varieties The diversity of a set of solutions P should increase when adding an individual p not yet in P. This fundamental property assures that additional solutions increase the diversity, i.e. it assures that increased species richness is reflected in the diversity measure.

Twinning Diversity should stay constant when adding an individual p already in P. Intuitively, if diversity is understood as the coverage of a space by a set of solutions, adding duplicates should not increase the coverage.

Monotonicity in Distance If all pairs of solutions in a population P are at least as dissimilar (measured by d) as those in another population P', the diversity of P should not be smaller than the diversity of P'.

It has been found that the measure proposed by Solow and Polasky [18] fulfills the requirements best and can be computed with reasonable computational complexity.

The Solow-Polasky measure $D(P)$ of a population $P \subseteq \mathcal{X}$ is determined as follows: Suppose P contains the n solutions $p_1, ..., p_n$ where $|P| = n$. Furthermore, $d(p_i, p_j)$ denotes the distance between solutions p_i and p_j. Then we can define the (n, n)-matrix $M = (m_{ij})$ with elements

$$m_{ij} = \exp(-\theta \cdot d(p_i, p_j)) \quad \text{for all } 1 \leq i, j \leq n$$

Then, the Solow-Polasky measure can be given as

$$D(P) = eM^{-1}e^T$$

where $e = (1, 1, \ldots, 1)$ and e^T denotes its transpose. In other words, $D(P)$ is the sum of all matrix elements of M^{-1}.

The Solow-Polasky measure yields real values in the interval $[1, |P|]$, which can be interpreted as the number of different species found in the population, where individuals which lie close to each other belong to the same species. The parameter θ normalizes the relationship between distance d and the number of species. As the selection of a distance d is problem domain specific, the value of θ has to be appropriately set. Following our experimental evaluations, the choice of θ is not critical as long as the matrix elements of M are in a reasonable interval, i.e. $10^{-5} \leq m_{ij} \ll 1$, $\forall i, j$, $i \neq j$.

3.2 Diversity-based Selection

In the NOAH algorithm, the diversity measure is used as a selection criterion, see Algorithm 4. According to Eq. (1), the operator selectDiv should preferably select the subset $P'' \subset \{P \cup P'\}$ with n elements which maximizes the diversity

$$D(P'') \geq D(P''') \quad \text{for all } P''' \subset \{P \cup P'\}, |P'''| = n$$

where $P'' = \text{selectDiv}(\{P \cup P'\}, n)$.

As testing all possible subsets is infeasible due to combinatorial explosion, we suggest to use the usual greedy strategy which removes one solution after another from the population P until only n solutions remain. In each step, the solution which contributes least to the diversity is discarded. Here, the contribution of a solution $p \in P$ to the diversity of the set P is defined as $D(P) - D(P\backslash\{p\})$, i.e. the difference between the diversity of the whole set and the diversity of the set without the solution p.

The computational complexity of the calculation of an optimized subset in selectDiv is now determined by the fact that we have to remove n solutions and for each of them, we have to test between $n + 1$ and $2n$ candidates. Each candidate evaluation for p necessitates the computation of $D(P\backslash\{p\})$ whose complexity is dominated by the matrix inverse calculation, which is $\mathcal{O}(n^3)$.

As a result, the computational complexity of selectDiv is reduced to $\mathcal{O}(n^5)$ in comparison to an exponential complexity, while giving up on the optimality of the obtained subset. Unfortunately, the computational complexity still is unacceptable for practical purposes, i.e. large population sizes. The next subsection describes an improved algorithm which reduces the complexity to $\mathcal{O}(n^3)$.

3.3 Fast Diversity-based Selection Algorithm

As described above, the complexity of $\mathcal{O}(n^5)$ to determine an optimized subset with maximal diversity is still a serious performance bottleneck. In the following we therefore suggest a novel way to (a) calculate the contributions of solutions to the Solow-Polasky measure and (b) to update the measure after removing a solution which only requires one matrix inversion in the whole selection process, therefore reducing its complexity to $\mathcal{O}(n^3)$.

First, we provide some definitions and known relations from linear algebra which will be used. Assume that we have a symmetric matrix M and its inverse M^{-1} which are partitioned in the following form:

$$M = \begin{pmatrix} A & b \\ b^T & c \end{pmatrix}, \quad M^{-1} = \begin{pmatrix} \bar{A} & \bar{b} \\ \bar{b}^T & \bar{c} \end{pmatrix}$$

where c and $\bar{c}$ are single elements, b and $\bar{b}$ are column vectors and b^T and $\bar{b}^T$ denote their transpose. We also make use of the notion $\Sigma(M) = \sum_{i,j} m_{i,j}$ which is the sum of all elements of the matrix M. Finally, we use the well known result for the block matrix inverse of M:

$$A^{-1} = \bar{A} - \frac{1}{\bar{c}} \cdot \bar{b} \cdot \bar{b}^T$$

We now want to calculate the contribution of a single solution to the Solow-Polasky measure. Remember that the Solow-Polasky measure is the element-wise sum of the inverse M^{-1} of the transformed pairwise distance matrix M of all solutions, i.e. $D(P) = \Sigma(M^{-1})$. Note that M can be described in the partitioned form as M is symmetric due to the symmetry of the distance measure, i.e. $d(p_i, p_j) = d(p_j, p_i)$ for all $p_i, p_j \in \mathcal{X}$.

If a solution is discarded from P, its corresponding row and column are deleted from the distance matrix M. Assume without loss of generality that the solution we want to discard corresponds to the last row and column of M, i.e. we want to delete the last row and the last column from M and determine the impact on the Solow-Polasky measure. This difference in the measure can now be calculated as follows:

$$\begin{aligned}
\Sigma(M^{-1}) - \Sigma(A^{-1}) &= [\Sigma(\bar{A}) + 2\Sigma(\bar{b}) + \bar{c}] \\
&\quad - [\Sigma(\bar{A}) - \frac{1}{\bar{c}}(\Sigma(\bar{b}))^2] \\
&= \frac{1}{\bar{c}}[2\bar{c}\Sigma(\bar{b}) + (\bar{c})^2 + (\Sigma(\bar{b}))^2] \\
&= \frac{1}{\bar{c}}(\Sigma(\bar{b}) + \bar{c})^2
\end{aligned}$$

The term $\frac{1}{\bar{c}}(\Sigma(\bar{b}) + \bar{c})^2$ can be interpreted as the normalized squared sum of the last column's elements of M^{-1}. By comparing all of these terms we can determine the solution which leads to the least difference in the diversity measure by $\mathcal{O}(n^2)$ operations.

Afterwards, we have to delete from M the solution with the smallest contribution and set the new distance matrix M' to the corresponding submatrix. If we again suppose without loss of generality that the solution with the smallest loss in diversity was associated to the last column, we have $M' = A$. In order to repeat this process for further solutions we would have to determine the inverse $M'^{-1} = A^{-1}$ which would need $\mathcal{O}(n^3)$ computations in a naive implementation. But using the above results on block matrix inverses, we can reduce this computation to $\mathcal{O}(n^2)$ computations. As a result, the removal of one element needs $\mathcal{O}(n^2)$ computations which leads to the desired $\mathcal{O}(n^3)$ complexity for the whole subset computation in selectDiv.

4. EXPERIMENTS

4.1 Evaluation and Comparison

In this section we compare NOAH to several other standard evolutionary algorithms with and without diversity preservation mechanisms. The purpose of this experimental evaluation is to see whether the considered set of algorithms is able to reach a given barrier, and if so, what conclusion can be drawn about the diversity of the final populations.

Name	Diversity Preserving	Mating Selection	Environmental Selection
NOAH	yes	Random without replacement	see Section 2
DetC	yes	Random without replacement	Deterministic crowding [14]
ResT	yes	Random with replacement	Restricted tournament [11]
Diff	yes	Random without replacement	Diffusion model [9]
Clear	yes	Random without replacement	Clearing procedure [16]
Share	yes	Fitness sharing [10, 9]	Pairwise tournament
Tour	no	Random without replacement	Pairwise tournament
Random	no	n/a	n/a

Table 1: Compared algorithms.

	NOAH		DetC		ResT		Diff		Clear		Share		Tour		Random	
3-Sat $v = 2$	22	4.2073	28	1.8041^+	30	3.6416	30	3.5026	30	3.6582	28	1.1309^+	30	1.3328^+	0	NaN
3-Sat $v = 5$	29	6.6118	30	4.8804^+	30	4.6674^+	30	4.7193^+	30	4.6985^+	30	1.726^+	30	1.6693^+	9	1.5593^+
3-Sat $v = 10$	30	7.0279	30	6.1889^+	30	5.272^+	30	5.3353^+	30	5.3159^+	30	2.306^+	30	1.9727^+	27	6.9609
nk-L. $v = 23$	10	1.1847	0	NaN	0	NaN	0	NaN	0	NaN	1	1	1	1.023	0	NaN
nk-L. $v = 25$	30	2.2627	4	1	7	1	7	1.1227	6	1	8	1.0057	7	1.0033	0	NaN
nk-L. $v = 30$	30	6.9834	30	6.1128^+	30	6.0892^+	30	6.0347^+	30	6.1159^+	30	1.1042^+	30	1.1273^+	0	NaN

Table 2: Experiment results of 30 runs. Columns show the different algorithms, rows the different problems (with the corresponding barrier value v). For each problem/barrier value pair and each algorithm there are two values, where the left one is the number of runs that had at least one solution on the barrier, and the right number is the mean diversity of the solutions that reached the barrier. A $+/-$ beside the diversity means that the diversity of NOAH is significantly better/worse than the diversity of that particular algorithm.

Optimization Problems

For the comparison, we selected two well-known test problems: The nk-Landscapes problem [13] and the 3-Sat problem [15]. In the nk-Landscapes problem, there are n decision variables (in our case, $n = 100$). Each decision variable is influenced by k (in our case $k = 10$) randomly chosen other decision variables. The decision variables are binary, i.e. they can either take the value 0 or 1. Each decision variable together with the influencing decision variables codes an index in a randomly generated fitness matrix. The overall fitness then is the sum of the fitness values coded by each decision variable.

The 3-Sat problem is a specific Boolean satisfiability problem. In our case, the Boolean expression which has to be satisfied consists of 200 clauses with 3 elements each. A clause is true if any of its elements is set to one, and the whole expression is true if all clauses are true. As an objective function, we use the number of false clauses, leading to a minimization problem which has an optimal value of 0 (which can only be reached if the expression is satisfiable). Our problem has 50 decision variables, where each clause contains 3 randomly selected decision variables as its elements.

Both optimization problems that we consider have binary search spaces. We here suggest to use the Hamming distance between decision vectors as a distance measure. For example considering the 3-Sat problem, we want not only to be able to find out whether the expression is satisfiable, but also to find a whole set of assignments that satisfy the Boolean expression. These assignments should be as diverse as possible in terms of differing decision variables.

As a variation operator, we first apply a two-point crossover with probability 0.5. Then, each solution undergoes a one-point bitflip mutation, i.e. one of its (binary) decision variables is selected at random and set to its inverse value (1 instead of 0 and vice versa).

Compared Algorithms

All algorithms that we compare are listed in Table 1. Mating selection denotes the step where the parents that will be recombined and mutated are selected. During environmental selection the individuals which are to survive (from the pool of parents and offspring) are chosen. The NOAH algorithm optimizes according to Algorithm 1, with parameters $n = 20, g = 20, r = 10, c = 10$, and with pairwise tournament for selectObj in Algorithm 2. All algorithms use a population size of 20.

During deterministic crowding, offspring are generated by recombining and mutating 2 parents, and then, a pairwise tournament between each offspring and its more similar parent takes place, see [9]. In restricted tournament, offspring are generated in a standard manner, and then each offspring replaces the most similar parent, if it is better than said parent. In the Diffusion Model Evolutionary Algorithm, the solutions are located on a grid in a fixed manner, where each solution has 8 neighbors. During variation, each individual is recombined with one of its neighbors. The offspring which is more similar to the neighbor replaces the current individual if it is better. The Clearing Procedure generates offspring in a standard way. Then, it performs a pruning on the offspring in order to find the κ best individuals in each niche. Niches are defined by a parameter σ, which in our case is set to 0.2 for all problems. Also, we use $\kappa = 1$, i.e. we use only one representative per niche. This representative then replaces the most similar parent, if it is better than that parent. When using fitness sharing, the fitness of each individual is decreased prior to selection, depending on the closeness and number of neighbors. Random selection with replacement is just a random selection of parents, where each individual can be selected multiple times. In the same selection without replacement, each individual can only be selected once. In pairwise tournament, pairs of solutions are selected and the better one is kept. Finally, the random algorithm simply generates random solutions and keeps the 20 best ones (if more than 20 individuals have the same best value, the most diverse ones are kept).

Experimental Setup

We test each problem with different barrier values. We compare the number of runs that achieved at least one solution with the barrier value, and the diversity of the solutions that reached the barrier value. In order to be able to fairly compare the different algorithms, all objective values below the barrier are set to the barrier, such that there is no selection pressure below the barrier. This way, the algorithms are free to optimize the diversity of the population after the barrier has been reached.

For each problem, the number of objective function evaluations *fEvals* is fixed. Note that NOAH terminates as soon as its bound reaches the barrier value, or when *fEvals* function evaluations have been performed, whichever happens first. For the nk-Landscapes and the 3-Sat problem, *fEvals* was set to $5 \cdot 10^5$ and $3 \cdot 10^6$, respectively.

The algorithms were run 30 times on each problem/barrier value pair. To test the resulting diversity values for significant differences, a Kruskal-Wallis test as described in [4] has been carried out, using the Conover-Inman procedure, Fisher's least significant difference method performed on ranks and a significance level of 1%.

Results

The results are shown in Table 2. For the two higher barrier values of 3-Sat, the algorithms mostly reach the barrier value, and the resulting diversity of NOAH is always significantly better than that of the other algorithms (except random search). For the lowest barrier value, NOAH sometimes does not reach the barrier, which can be explained with the fact that it spends a considerable amount of function evaluations on diversity optimizations. Furthermore, the diversity of NOAH is only significantly better than that of $DetC$, $Tour$ and $Share$, whereas there is no significant difference to the diversity of $ResT$, $Diff$ and $Clear$. This can be explained by the fact that when the barrier value is low, only small parts of the decision space are on or below the bound, and therefore diversity cannot be optimized as much as if the barrier was higher.

For the nk-Landscapes problem it is interesting to note that for the lowest barrier value, most algorithms cannot reach that barrier (except $Tour$ and $Share$, which reach the barrier once). This is in contrast to NOAH, which reaches the barrier every third time. This indicates that diversity might help identifying the global optimum by covering as many local optima as possible. This can still be seen for the second lowest barrier value, which is always reached by NOAH, whereas it is only reached in about 24% of the cases by the best other algorithms ($Tour$ and $ResT$). For the highest barrier value, all algorithms always reach it (except random search), but NOAH's diversity is always significantly better than the other algorithms diversity.

4.2 Bridge Construction

In order to qualitatively interpret the simultaneous optimization of the objective function and the set diversity (mixed multiobjective optimization), we applied the algorithm NOAH to a more realistic problem. Here, we would like to see whether truss bridges constructed and optimized by NOAH 'look' more diverse than bridges produced by a standard evolutionary algorithm.

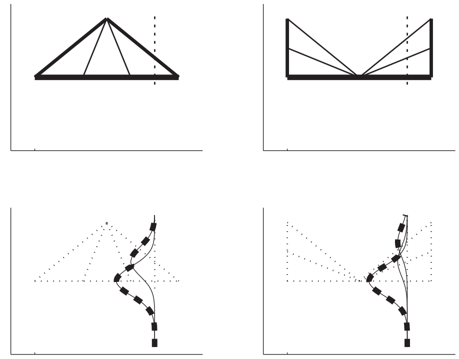

Figure 2: Distance calculation between two bridges (upper row, the first/second bridge is in the first/second column). The Gaussian curve calculation is shown for a specific vertical slice (dotted line in the upper row). For each crossing connection, a Gaussian is drawn (solid line in lower row). The maximum of these Gaussians is then used for the distance calculation (dashed line in lower row).

Optimization Problem

As an optimization problem, we selected the bridge construction problem [1]. The goal is to build a truss bridge which is able to carry a given load and which is as 'cheap' as possible. Costs are computed by adding the necessary material to build the bridge (total length of connections multiplied by their cutting area). Except for the fixed main horizontal deck on which the load is applied, the bridge can be constructed arbitrarily, i.e. connections and nodes can be added, moved and removed. Note that there is only mutation and no recombination in the variation operator for the bridge optimization, see [1].

As mentioned in Section 1.1, a distance measure in decision space is needed. But how can the distance between two bridges be measured? We found that the following distance measure which is also depicted in Figure 2 yields good results: Both bridges are cut into a fixed number of vertical slices. For each slice, there are certain connections of different widths (that correspond to the cross-section area of the connection) which cross that slice. For each connection, we draw a Gaussian with a given variance (in our case 5) and the mean at the point where the connection cuts the slice, multiplied with the thickness of the connection. For each point on the slice we then take the maximum of the Gaussians of all connections that cut the slice, which gives us one curve per bridge. The difference between these curves of two bridges, summed up over all vertical slices, determines the distance of the two bridges. For an illustration see Figure 2. It can be seen that the slice shown in the Figure does not contribute much to the distance of the two bridges, as those two bridges both have the same largest deck (the thick horizontal deck).

Experimental Setup

As the optimum bridge is not known, we first run a a standard evolutionary algorithm that uses pairwise tournament

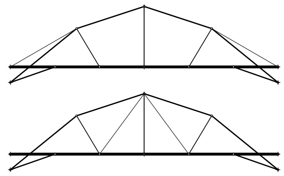

Figure 3: Upper: Best bridge optimized with standard EA. The horizontal deck is fixed, and carries the load which the bridge must be able to sustain. Lower: Most distant bridge to upper bridge, also optimized with the standard EA.

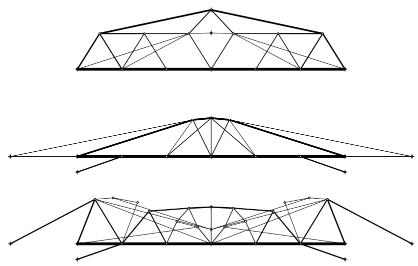

Figure 4: Three diverse bridges determined by NOAH using a barrier objective value which is 20% higher than the cost of an EA-optimal bridge.

for 200'000 function evaluations and with a population size of 20. Then we optimized the same bridge problem with NOAH, where we set the barrier value to 20% above the cost of the best bridge found by the standard evolutionary algorithm. NOAH was run with parameters $n = 20, g = 10, r = 10, c = 5$. Note that NOAH's bound reached the barrier after 19'000 function evaluations.

Results

The best bridge found by the standard evolutionary algorithm as well as the bridge with the largest distance to this best bridge in the population[2] is depicted in Figure 3. As can be seen, these two bridges are very similar, hence there is not much diversity in the population. Figure 4 on the other hand shows three diverse bridges found by NOAH. These bridges are visually much more different than the bridges found with the standard EA and provide the decision maker with multiple alternative bridges that can carry a given load and that do not cost more than 20% of the cheapest bridge found by a standard evolutionary algorithm.

5. CONCLUSIONS AND OUTLOOK

This paper proposes a method to generate a set of maximally diverse solutions which are better in terms of objective value than a certain fitness value. All solutions beyond this barrier are supposed to be acceptable to the decision maker.

To this end, we propose an algorithm called NOAH which alternates between optimizing the population for diversity and for objective value, and which uses an adaptive constraint to ensure the quality of the solutions. Also, a new algorithm has been described that substantially reduces the computational complexity for the diversity optimization.

NOAH is compared to standard evolutionary algorithms with and without diversity preservation on the nk-Landscapes and the 3-Sat problem. It could be seen that NOAH achieves most of the time a significantly better diversity than the other algorithms, and never worse. On the nk-Landscapes problem it appears that the diversity preservation helps identifying better local optima, as NOAH achieves better fitness values than the other algorithms. Finally, NOAH was applied to a truss bridge construction problem. It was able to determine diverse bridges if 20% more costs are allowed than the cost of the best solution found with a standard evolutionary algorithm.

An important feature of NOAH is its ability to adaptively reduce its current bound value during optimization. In the future, it would be desirable to automatically tune the parameters of NOAH, especially the number of generations for which the optimization of fitness values takes place, as this parameter decides on the tradeoff between diversity and fitness optimization speed.

[2]Note that in the final population we only consider bridges that cost up to 20% more than the best bridge. Due to the stochastic nature of the selection process, the population can contain a few very expensive bridges, which we do not want to consider here.

6. REFERENCES

[1] J. Bader. *Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods*. PhD thesis, ETH Zurich, Switzerland, 2010.

[2] D. Beasley, D. Bull, and R. Martin. A sequential niche technique for multimodal function optimization. *Evol. Comput.*, 1:101–125, 1993.

[3] L. Bui, J. Branke, and H. Abbass. Multiobjective optimization for dynamic environments. In *CEC*, 2005.

[4] W. J. Conover. *Practical Nonparametric Statistics*. John Wiley, 3rd edition, 1999.

[5] K. A. de Jong. *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. PhD thesis, 1975.

[6] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2001.

[7] K. Deb and D. E. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Third international conference on Genetic algorithms*, 1989.

[8] K. Deb and S. Tiwari. Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization. *EJOR*, 185(3):1062–1087, 2008.

[9] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.

[10] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Second International Conference on Genetic algorithms and their application*, 1987.

[11] G. Harik. Finding multimodal solutions using restricted tournament selection. In *Sixth International Conference on Genetic Algorithms*, 1995.

[12] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments-a survey. *Evol. Comput.*, 9(3):303 – 317, 2005.

[13] S. A. Kauffman. *Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, 1993.

[14] S. W. Mahfoud. Crowding and preselection revisited. In *PPSN*, 1992.

[15] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of sat problems. In *AAAI*, 1992.

[16] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *IEEE International Conference on Evolutionary Computation*, 1996.

[17] A. Saha and K. Deb. A bi-criterion approach to multimodal optimization: Self-adaptive approach. In *SEAL*, 2010.

[18] A. R. Solow and S. Polasky. Measuring bilological diversity. *Environmental and Ecological Statistics*, 1:95–103, 1994.

[19] T. Ulrich, J. Bader, and L. Thiele. Defining and optimizing indicator-based diversity measures in multiobjective search. In *PPSN*, 2010.

[20] X. Yu and M. Gen. *Introduction to Evolutionary Algorithms*. Springer, 2010.