NC'2000 - regular paper

Evolution Strategy with Neighborhood Attraction

Jutta Huhse, Andreas Zell
Lehrstuhl Rechnerarchitektur
Wilhelm-Schickardt-Institut für Informatik
Eberhard-Karls-Universität Tübingen
Köstlinstr. 6, D-72074 Tübingen, Germany
E-mail: {huhse,zell}@informatik.uni-tuebingen.de
Fax: (+49/0)7071-92 29 83

Abstract
A new combination of self-organizing maps and evolution strategies (ES) - the evoution strategy with neighborhood attraction (EN) - is described in this paper. Performance tests with the new strategy and comparisons with existing ES led to a reduced form of EN with small populations which was able to outperform comparable ES or to achive similar performance.

Keywords
Evolution strategy, neighborhood, recombination, SOM, small sized populations

1 Introduction

Besides genetic algorithms (GA) and evolutionary programming (EP), evolution strategies (ES) belong to the main stream of evolutionary algorithms (EA). Developed in the late 1960s by Rechenberg and Schwefel and improved in the following time [Rechenberg 94], [Schwefel 95], [Bäck et al. 97] their main application is the optimization of real-valued multi-parameter problems. In contrast to other search- or optimization strategies, ES do not require any restrictions of the search space such as continuity, unimodality or existence of any derivations. They work directly on information about the quality of a potential solution of the function to be optimized. Therefore, they are more generally applicable. Other interesting problem solvers are artificial neural networks. They may be viewed as a black box, finding the appropriate output for every input vector [Hertz et al. 91]. Since the performance of ES and neural networks is dependant on several parameters whose optimal settings are not known a priori, it is evident to apply one strategy to the other. Some research has been done in using EAs for optimizing neural networks (e. g. [Montana et al. 89], [Janikow et al. 91], [Koza et al. 91]). The strategy presented here works the other way round, i. e. it utilizes features of neural networks on evolution strategies.
A special class of neural networks - the self-organizing maps (SOM) - were developed in the 1980s by Kohonen [Kohonen 95]. They are used to find a topology-preserving mapping from the input space to the output space. This is achieved by spreading the weight vectors of the neurons over the entire input space.
This paper pursues the concept of using this ability of the SOM to search the problem space of an ES to find the optimum.
The following chapter depicts the basics of evolution strategies and self-organizing maps. Chapter 3 describes the new evolution strategy with neighborhood attraction. Then the chapter 4 lists the test functions which were used to evaluate the new strategy. The test series and the results are presented in chapter 5. Chapter 6 contains a discussion and ideas for further research on the EN. The final chapter concludes with a short summary of this work.

2 Basics of Evolution Strategies and Self-Organizing Maps

The idea for the EN arose from the combination of artificial neural networks (especially SOM) and evolution strategies. Therefore, both of these topics will be regarded in the following sections.
Evolution strategies (ES) are mostly used for optimization of real-valued multi-parameter problems. They work on a population P of potential solutions (*individuals a*) by manipulating these individuals with genetic operators such as mutation and recombination.
The population consists of μ *parent individuals* a_{Pm} $(m = 1 \ldots \mu)$. Each individual consists of a vector of object variables $\vec{x} = (x_1, x_2, \ldots, x_n)$ $(x_i \in \Re)$ (where $d=n$ is the dimension of the optimization problem) and of strategy variables. The μ parent individuals a_{Pm} produce λ $(\lambda \geq \mu)$ offspring individuals a_{Ol} $(l = 1 \ldots \lambda)$ by mutation and recombination.
Mutation is the most important operator in ES. It works by adding a vector of normally distributed random

variables $\vec{z}$ $(z_i = N(0, \sigma_{Oi}), i = 1 \ldots n)$ to the object variables vector $\vec{x}_P$ of the parent individual. The mutation step sizes σ_{Oi} are deduced from the parent's mutation step sizes σ_{Pi} (stored as strategy variables vector in a_P). There are different possibilities to calculate these values, e. g. standard adaptation or covariance matrix adaptation (for details, see [Bäck 96], [Hansen et al. 96]). The object variables of the offspring are: $x_{Oi} = x_{Pi} + z_i$
Recombination is another genetic operator. It works on two or more individuals in several variations (see [Bäck 96]). Here we explain the (generalized) intermediate recombination, performed on two parent individuals a_{P1} and a_{P2}:

$$x_{Oi} = x_{P1i} + \chi(x_{P1i} - x_{P2i}), \qquad \chi \in [0, 1] \qquad \textit{equ. 2.1}$$

The object variables vector $\vec{x}_O$ of the offspring individual is generated by recombining the elements of the object variables vectors $\vec{x}_{P1}$ and $\vec{x}_{P2}$ of the two parents.
If $\chi = 0.5$, this is called *intermediate recombination*, and the offspring is placed right in the middle between the two parents.
Self-organizing maps (SOM) - also known as „Kohonen"-maps - are a special class of artificial neural networks which perform a topology-preserving mapping from a *n*-dimensional input space to a *m*-dimensional output space. The neural network consists of one layer of *K* active neurons with a neighborhood relationship defined among them. A neighborhood relationship frequently used in literature is a two-dimensional, quadratic grid where each neuron is associated a fixed grid position. Neighborhood is defined by a direct connection between two neurons.
To obtain the topology-preserving mapping, an input vector $\vec{x} = (x_1, x_2, \ldots, x_n)$ is chosen from the input space and compared to the weight vectors $\vec{w}_k = (w_{k1}, w_{k2}, \ldots w_{kn})$ $(k = 1 \ldots K)$ of all *K* neurons. Neuron *c*, whose weight vector $\vec{w}_c$ is most similar (or has the smallest distance) to the input vector, is called the winner neuron. For learning, the weights of neuron *c* are changed in direction to the input vector. According to the defined neighborhood relationship, the weight vectors of the neurons *j* neighboring neuron *c* are also adapted. Adaptation strength of $\vec{w}_j$ decreases with increasing distance between the neighbor *j* and the winner *c*. Equation 2.2 expresses these relationships.

$$\vec{w}_j(t+1) = \vec{w}_j(t) + \eta(t) h_{cj}(t)[\vec{x}_j(t) - \vec{w}_j(t)] \qquad \textit{equ. 2.2}$$

t stands for discrete time steps for adaptation, $h_{cj}(t)$ is the neighborhood function which may alter with time, and $\eta(t)$ is the learning rate.

3 Evolution Strategy with Neighborhood Attraction

The idea behind *E*volution strategies with *N*eighborhood attraction (EN) is to transfer the neighborhood and the learning rule defined for SOM-neurons onto the individuals of an ES.
The individuals which are unorganzied in convential ES now (in the EN) have neighborhood relations. The neighborhood between the μ parent individuals is constituted by arranging them in an orthogonal, elastic grid. As described for the SOM, each individual can be identified by its fixed grid position, and two individuals are neighbors if they are directly connected on the grid (see fig. 3.1 left).

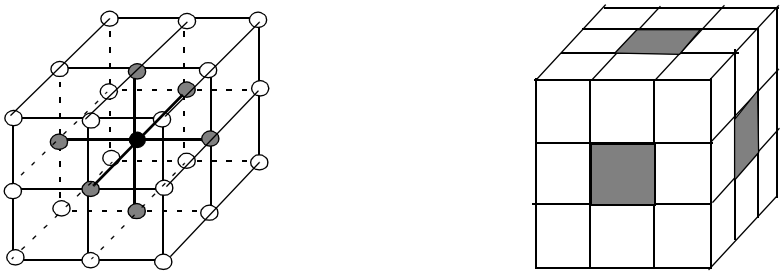

Fig. 3.1: Left: Neighborhood grid between the parent individuals; all grey individuals are neighbors of the black individual. Right: Division of the problem space into hypercubes; during initialization one individual is placed randomly into each hypercube

In contrast to conventional ES and SOM the initial values of the object variables of the EN- individuals are not assigned randomly. Rather, the problem space is divided into equally sized hypercubes, each of them corresponding to one grid position (see fig. 3.1, right). The object variables of the associated individual are initialized with equally distributed random values within the ranges of its hypercube.
As is customary in ES, the EN-individuals are evaluated using the fitness function.
The newly introduced genetic operator - the neighborhood attraction - manipulates the EN-individuals according to one learning step in a SOM. Every individual a_P is attracted to its best neighbor a_{Nb} and thus becomes the

offspring a_O. The object variables $\vec{x}_O$ are calculated with equation 3.1, the neigbourhood relations are retained unchanged.

$$\begin{aligned} \vec{v}_{mut} &= \vec{x}_{Nb} - \vec{x}_P \\ s_{eff} &= \delta \qquad\qquad \delta \in (0, 1) \\ \vec{x}_O &= \vec{x}_P + s_{eff} \cdot \vec{v}_{mut} \end{aligned} \qquad \textit{equ. 3.1}$$

$\vec{v}_{mut}$ is the difference vector between the object variable vectors of the parent $\vec{x}_P$ and its best neighbor $\vec{x}_{Nb}$. The parameter δ defines the strength of the attraction along the difference vector and $\vec{x}_O$ denotes the object variables of the offspring. (The identifiers $\vec{v}_{mut}$ and s_{eff} are used to make a comparision with equ. 3.2 easier.) If the parent individual a_P is considered better than all its neighbors a_{Nj} ($j = 1 \ldots g$, g is the number of neighbors) a "simple conventional" mutation (referred to as *ES-mutation* here) is performed. λ offsprings are generated according to equation 3.2.

$$\begin{aligned} \vec{v}_{mut,l} &= \overrightarrow{N(0, 1)} \qquad\qquad l = 1 \ldots \lambda \\ d_{min} &= min\{\left\|\vec{x}_P - \vec{x}_{Nj}\right\|\} \qquad j = 1 \ldots g \\ s_{eff} &= \frac{1}{n} d_{min} \\ \vec{x}_{Oi} &= \vec{x}_P + s_{eff} \cdot \vec{v}_{mut,i} \qquad i = 1 \ldots n \end{aligned} \qquad \textit{equ. 3.2}$$

(n is the number of object variables.)
The effective step size s_{eff} is determined by the reciprocal number of object variables[1] and by the distance d_{min} to the nearest neighbor. Thus, a mutation which jumps over a neighbor and an entanglement of the grid becomes less likely. During the contraction of the grid the effective step sizes decrease due to the influence of d_{min}.
Out of the perspective of evolution strategies the neighborhood attraction can be categorized somewhere between a mutation and a recombination. The mutative aspect exists since every single parent individual is manipulated, and, starting from that individual, an offspring is placed in the neighborhood. However, the placement does not happen on the surfaces of hyper-ellipsoids, rather along a predetermined direction (vector between the parent and its best neighbor), and the length of that mutation step is derived from the effective stepsize s_{eff}. The recombinatorical aspect is given because the neighborhood attraction is no asexual operator since it uses information about the neighbors of the parent individual. If the best neighbor is determined, essentially a generalized intermediate recombination is performed (see equ. 2.1), however with χ being not a random number but being calculated as s_{eff}. Differences between EN and ES are also in the selection method. The difficulty for the EN is caused by the neighborhood relation. In order to avoid having to recalculate this for every generation, the selection is not performed on all offsprings of the whole population (+ all the parents) in parallel to determine the μ best individuals. Rather every parent individual (together with its λ offsprings) is judged on its own, i. e. one selection per grid-position is done. The selected individual then replaces the parent individual; the grid position and the neighborhood are taken over (s. a.). During neighborhood attraction, the parent is always replaced by its offspring, even if its fitness is worse. After an ES-mutation, a plus selection will be done among the parent and its offsprings.

4 Test Functions

The functions below (for details, see Appendix) have been used as a test bed to investigate the EN-strategy. They are numbered according to [Bäck 92] and [Hansen et al. 95].
The test functions were selected on the basis that uni-modal (f_1, f_2, f_6, f_{15}, q_1) and multi-modal functions (f_7, f_9, f_{21}), symmetric and non-symmetric functions (f_2), easily optimized functions (e. g. f_1) and rather difficult ones (e. g. f_7) were represented.

- f_1 Sphere model
- f_2 Generalized Rosenbrock's function
- f_6 Schwefel's Function

1. according to [Rechenberg 94], who proposes for his basic algorithm for a $(1^{+}_{,}\lambda)$-ES to make the length of the mutation vector independent of the number n of variables by generating the normally distributed vector elements with the mean zero and with the variance $\sigma = 1/\sqrt{n}$

- f_7 Generalized Rastrigin's function
- f_9 Ackley's function
- f_{15} Weighted Sphere model
- f_{21} Griewangk's funktion
- Q_1 Hyperellipsoid

5 Test Series

The test series can be divided into the basic tests performed on EN to investigate whether that strategy works at all and in tests for comparing EN to other evolution strategies.
To validate the EN, a new simulation program was developed at the Wilhelm-Schickardt-Institute for Computer Science, University of Tübingen. For comparison tests, the EvA-system for Evolutionary Algorithms [Wakunda et al. 97] (developed at the same institution) was used.

5.1 Basic Tests

The first test series was performed on EN to investigate if that strategy works at all. The test functions were used with the dimension $d = 2$, and the contraction of the individual-grid could be visualized (see figure 5.1).

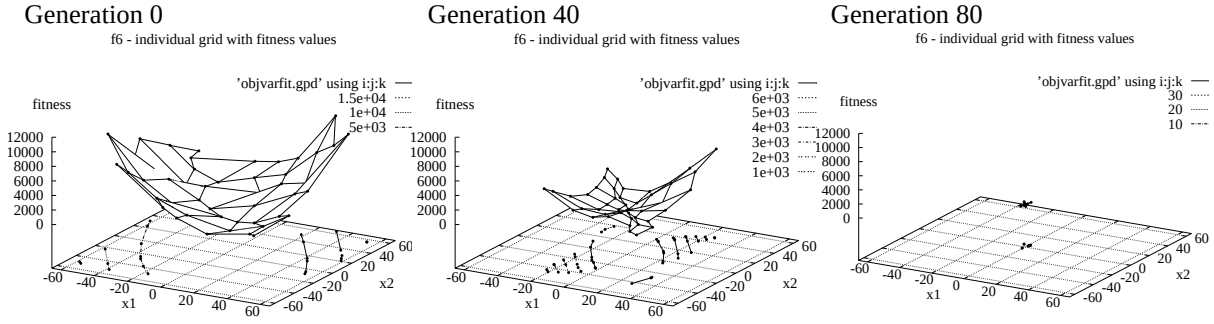

Fig. 5.1: Contraction of the individual grid, demonstrated on f_6 with μ =49, λ =2 and δ =0.1

The next step was to test ENs suitability in higher dimensions. The dimensions used were $d_1 = 10$ and $d_2 = 20$. In addition, the influence of the parameters μ and λ were studied. μ varied between 2 and 256, λ varied between 1 and 20. To evaluate the test series, the number of test function evaluations (abscissa) and best corresponding fitness value (ordinate) were graphically represented. The results were that EN was able to minimize most of the test functions (not f_7, but this function could not be optimized by es, rec_i, rec_{geni} either; see section 5.2) and that in most cases the parameter settings with small μ and λ ($\mu = 2$ and $\lambda \in [2, 5]$) were fastest.
In a further test series, the parameter δ (see equation 3.1) was examined. It was set to values ranging between 0.001 and 0.9. It was determined that δ is a very sensitive parameter which - in the extreme case - may decide on the success or failure of the EN. Too small settings lead to a very slow progress of the optimization, and too large settings result in a premature „grid-collaps" where all individuals are concentrated in the same non-optimal point. Good results were achieved for parameter settings $\delta \in [0.0075;0.03]$; $\delta = 0.01$ is a practical value for starting the search.

5.2 Comparision Tests

With the experiences gathered in the tests above, we proceeded to compare EN to other evolution strategies. These include the following:

- „Simple" Evolution Strategy (es)
- Evolution Strategy with Intermediate Recombination (rec_i)
- Evolution Strategy with Generalized Intermediate Recombination (rec_{geni})
- Evolution Strategy with Covariance Matrix Adaptation (cma)

„Simple" Evolution Strategy (es)
Analogous to the rather simple EN, a simple evolution strategy with comma-selection and uniform mutation step adaptation, but without recombination or covariance matrix adaptation, was decided on. For better comparison, the simple evolution strategy was run for a variety of different μ-λ-combinations, ranging in $\mu \in [1, 20]$ and

$\lambda \in [2\mu, 12\mu]$. Results are shown for the best (fastest) combinations, for a „standard-combination“ with $\mu = 15$ and $\lambda = 105$, and for the combination $\mu = 2$ and $\lambda = 4$ which gets closest to the parameter settings in EN (see fig. 5.2 - 5.7).

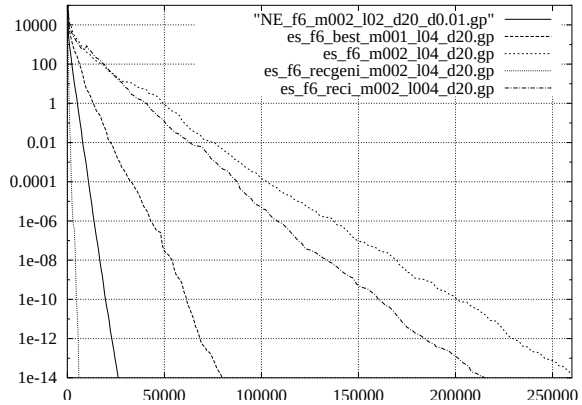

Fig. 5.2: Tests with function f_6

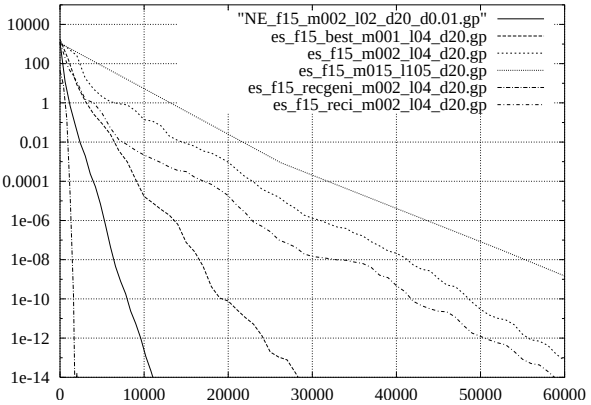

Fig. 5.3: Tests with function f_{15}

Evolution Strategy with Intermediate Recombination (rec_i)

Considering that the EN works well with $\mu = 2$, the neighborhood grid reduces to a simple connection between the only two individuals. The neighborhood attraction of one EN-individual to its better neighbor is a movement along the difference vector of these individuals.

Equation 2.1 shows the intermediate recombination after [Bäck 96]. If the population size is $p = 2$, the resemblance to EN with $\mu = 2$ and $\delta = 0.5$ is obvious.

Therefore, the test series were conducted with an evolution strategy with intermediate recombination, comma-selection, $\mu = 2$ and $\lambda = 4$ (see fig. 5.2 - 5.7).

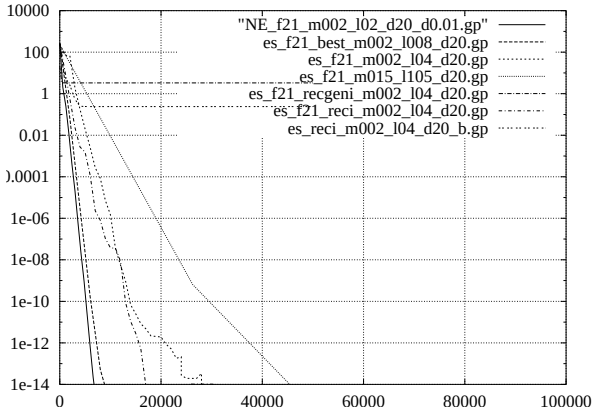

Fig. 5.4: Tests with function f_{21}

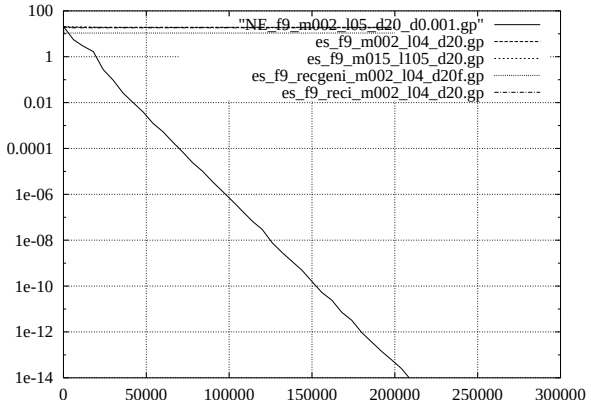

Fig. 5.5: Tests with function f_9

Evolution Strategy with Generalized Intermediate Recombination (rec_{geni})

The results of the basic tests with δ and of the comparison with rec_i led to this test series.

Bäck [Bäck 96] pointed out, that although for generalized intermediate recombination arbitrary weight factors from the interval [0, 1] besides the value 0.5 are allowed, this has not yet been extensively tested.

The findings on the importance of δ were transferred to the weight factor χ of the generalized intermediate recombination. The settings for the test series with generalized intermediate recombination were $\mu = 2$, $\lambda = 4$, comma-selection, uniform mutation step adaptation, intermediate recombination of both the object variables and the strategy variables and $\chi = 0.01$ (see fig. 5.2 - 5.7).

The better performance of rec_{geni} compared to rec_i confirms the importance of the weight factor χ and the neighborhood attraction factor δ.

Evolution Strategy with Covariance Matrix Adaptation (cma)

A last comparison was done between EN and cma. Unfortunately for EN, cma was considerably faster than EN on most functions tested, measured in function evaluations. This was to be expected, regarding the considerably more sophisticated, but also more time consuming, computations for computing offsprings in cma. cma belongs to the

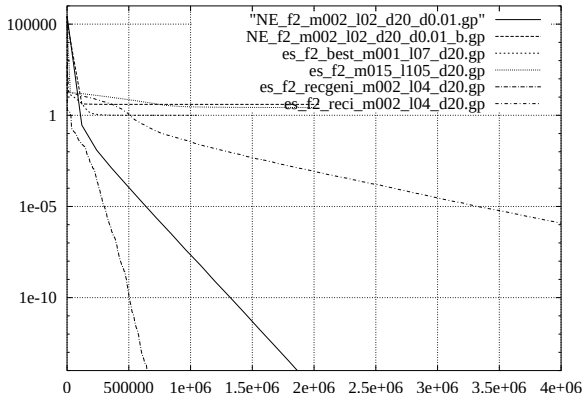

Fig. 5.6: Tests with function f_2

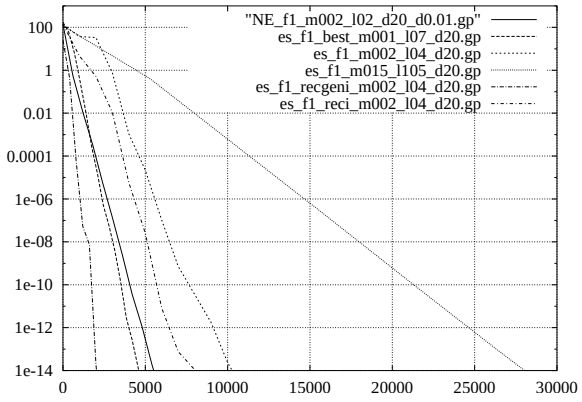

Fig. 5.7: Tests with function f_1

fastest evolution strategies, which uses an ingenious combination of different strategy-elements.

6 Discussion

The results for the basic tests have shown that the evolution strategy with neighborhood attraction is able to solve difficult optimization tasks.

The observation why small values for μ result in a better performance can be explained for the easy uni-modal test functions (e. g. f_1, f_6, f_{15}) by the small populations that suffice in conventional ES. Furthermore the selection method in EN (which was established to preserve the neighborhood) cannot be seen as a great benefit for uni-modal problems. Selection takes place for every grid position independently (see chapter 3). This leads to performance problems in a large grid in an uni-modal function since many individuals are selected into the next generation which do just have a poor fitness and are far away from the optimum (see fig. 5.1). This leads to a strongly reduced neighborhood grid and therefore to a strategy element that is more similar to recombination in ES than to learning in SOM.

For multi-modal functions like f_9 it is quite astonishing that a EN with $\mu = 2$ and $\lambda = 5$ is successful, while a common ES gets stuck in a local optimum (as expected) (see fig. 5.5). This demonstrates that the neighborhood attraction is quite a powerful tool.

The comparison test series with EN and rec_i and rec_{geni} show that the neighborhood attraction is a promising element for ES: EN is better than the „simple“ ES, although EN does not use explicit mutation step adaptation. EN is also better than a conventional ES with intermediate recombination (rec_i). The ES with generalized intermidiate recombination (rec_{geni}) is also better than rec_i. Because rec_{geni} uses an element quite similar to EN the good performance of rec_{geni} confirms the benefit of EN. The comparision of EN and rec_{geni} shows similar results. EN is missing the mutation step adaptation of rec_{geni} but it appears that its neighborhood attraction, which is slightly different from generalized intermediate recombination, is often stronger than that.

Further work will be done on different neighborhood relations and improved selection methods. It is also conceivable to incorporate the mechanism of EN into other existing, elaborated ES to improve these.

7 Summary

The new evolution strategy with neighborhood attraction (EN) was presented and compared to other ES on the basis of a number of test functions. It could be seen that the EN is capable of optimizing the test functions with a performance better than or resembling to the performance of most of the compared ES, even though EN does not use explicit mutation step adaptation. The neighborhood attraction is a promising element for evolution strategies which should be worth further investigation.

References

[Bäck 92] Thomas Bäck, A user’s guide to genesys 1.0, Technical report, University of Dortmund, Department of Computer Science, System Analysis Research Group

[Bäck 96a] Thomas Bäck, Evolutionary Algorithms in Theory and Practic, 1996, Oxford University Press Inc., New York

[Bäck 96b] Thomas Bäck, Evolution Strategies: An Alternative Evolutionary Algorithm, in: J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, D. Snyers (Eds.), Artificial Evolution: European Conference, AE 95, Brest, France, September 1995, Selected Papers 1996, Springer, Berlin

[Bäck et al. 97] Thomas Bäck, Ulrich Hammel, Hans-Paul Schwefel, Evolutionary Computation. Comments on the History and Current State, April 1997, IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, pp. 3-17

[Bayer 95] Harald F. Bayer, Über die Anwendung selbstorganisierender Karten, 1995, Dissertation, Universität Stuttgart, Fakultät Informatik, Institiut für parallele und verteilte Höchstleistungsrechner der Universiät Stuttgart.

[Fletcher et al. 63] R. Fletcher, M. J. D. Powell, A rapidly convergent descent method for minimization, Computer Journal, 6:163-168, 1963

[Fogel 94] David B. Fogel, An Introduction to Simulated Evolutionary Optimization, IEEE Transactions on Neural Networks, Vol. 5, No. 1, January 1994

[Fogel 95] David B. Fogel, Evolutionary Computation - Toward a New Philosophy of Machine Intelligence, 1995, IEEE Press, New York

[Fogel 98] David B. Fogel, Evolutionary Computation - The Fossil Record, 1998, IEEE Press, New York

[Hansen et al. 95] Nikolaus Hansen, Andreas Ostermeier, Andreas Gawelczyk, 1995, Über die Adaptation von allgemeinen, Koordinatensystem-unabhängigen, normalverteilten Mutationen in der Evolutionsstrategie: Die Erzeugendensystemadaption, Technical report, Technische Universität Berlin ftp://ftp-bionik.fb10.tu-berlin.de/pub/papers/Bionik/tr-02-95.ps.Z

[Hansen et al. 96] N. Hansen, A. Ostermeier, 1996, Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation, in: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC '96), pp 312-317, Nagoya, Japan. IEEE. ftp://ftp-bionik.fb10.tu-berlin.de/pub/papers/Bionik/CMAES.ps.Z.

[Hertz et al. 91] John Hertz, Anders Krogh, Richard G. Palmer, Introduction to the Theory of Neural Computation, 1991, Addison-Wesley; Redwood City, CA 94065

[Janikow et al. 91] Cezary Z. Janikow, Zbignew Michalewicz, An Experimental Comparison of Binary and Floating Point Representation in Genetic Algorithms, in: R. Belew und L. Booker (Eds.), 1991, Proceedings of the Fourth International Conference on Genetic Algorithms, San Diego, California, Morgan Kaufmann, San Mateo, California

[deJong 75] Kenneth de Jong, An Analysis of the behaviour of a class of genetic adaptive systems, 1975, Dissertation, University of Michigan, Diss. Abstr. Int. 36(10)

[Kohonen 95] Teuvo Kohonen, Self-Organizing Maps, 1995, Springer Verlag Berlin Heidelberg,

[Koza et al. 91] John R. Koza, James P. Rice, Genetic Generation of both the Weights and Architecture for a Neural Network, IJCNN - International Joint Conference on Neural Networks, Volume 2, pp. 397-404, IEEE/INNS, 1991, Juli, Seattle, WA

[Montana et al. 89] David J. Montana, Lawrence Davis, Training Feedforward Neural Networks Using Genetic Algorithms, IJCAI-89 - Eleventh International Joint Conference on Artificial Intelligence, Volume 1, Proceedings, 1989, August, Detroit, Michigan, USA

[Rechenberg 94] Ingo Rechenberg, Evolutionsstrategie '94, 1994, frommann-holzboog, Stuttgart, ISBN 3-7728-1642-8

[Rojas 93] Raul Rojas, Theorie der neuronalen Netze, 1993, Springer-Verlag Berlin Heidelberg

[Scholz 95] Michael Scholz, Topologiegenerierung für Neuronale Netze durch Genetische Algorithmen, Dissertation am FB 13, TU-Berlin, 1995, Berlin

[Schwefel 95] Hans-Paul Schwefel, Evolution and Optimum Seeking, 1995 John Wiley & Sons, New York

[Wakunda et al. 97] J. Wakunda, A. Zell, 1997, Eva - a tool for optimization with evolutionary algorithms, in: Proceedings of the 23rd EUROMICRO Conference, Budapest, Hungary

[Zell 94] Andreas Zell, Simulation Neuronaler Netze, 1994, Addison-Wesley (Deutschland) GmbH

Appendix A

A.1 f_1 Sphere Model

$$f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2$$

$$-5.12 \le x_i \le 5.12$$

$$min(f_1) = f_1(0, \ldots, 0) = 0$$

A.2 f_2 Generalized Rosenbrock's Function

$$f_2(\vec{x}) = \sum_{i=1}^{n-1} (100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$$

$$-5.12 \le x_i \le 5.12$$

$$min(f_2) = f_2(1, \ldots, 1) = 0$$

A.3 f_6 Schwefel's Function

$$f_6(\vec{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j^2 \right)^2 = x^T \mathbf{A} x$$

$$-65.536 \le x_i \le 65.536$$

$$min(f_6) = f_6(0, \ldots, 0) = 0$$

A.4 f_7 Generalized Rastrigin's Function

$$f_7(\vec{x}) = nA + \sum_{i=1}^{n} (x_i^2 - A\cos(\omega\, x_i))$$

$$A = 10; \qquad \omega = 2\pi$$

$$-5.12 \le x_i \le 5.12$$

$$min(f_7) = f_7(0, \ldots, 0) = 0$$

A.5 f_9 Ackley's Function

$$f_9(\vec{x}) = -a \cdot \exp\left(-b \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(c x_i)\right) + a + e$$

$$a = 20; \qquad b = 0.2; \qquad c = 2\pi$$

$$-32.768 \le x_i \le 32.768$$

$$min(f_9) = f_9(0, \ldots, 0) = 0$$

A.6 f_{15} Weighted Sphere Model

$$f_{15}(\vec{x}) = \sum_{i=1}^{n} i \cdot x_i^2$$

$$-5.12 \le x_i \le 5.12$$

$$min(f_{15}) = f_{15}(0, \ldots, 0) = 0$$

A.7 f_{21} Griewangk's Funktion

$$f_{21}(\vec{x}) = 1 + \frac{1}{d} \cdot \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \left(\frac{x_i}{\sqrt{i}} \right)$$

$$d = 200$$

$$-100.0 \le x_i \le 100.0$$

$$min(f_{21}) = f_{21}(0, \ldots, 0) = 0$$

A.8 Q_1 Hyperellipsoid

$$Q_1(\vec{x}) = \sum_{i=1}^{n} \left(1000^{\frac{i-1}{n-1}} \cdot x_i \right)^2$$

$$-\infty \le x_i \le \infty$$

$$min(Q_1) = Q_1(0, \ldots, 0) = 0$$