Evolution Strategies, Random Network Keys, and the One-Max Tree Problem

Barbara Schindler and
Franz Rothlauf

Working Paper 8/2001
November 2001

Working Papers in Information Systems
Editor: Prof. Dr. Armin Heinzl

University of Bayreuth
Chair of Information Systems
Universitaetsstrasse 30
D-95440 Bayreuth, Germany
Phone +49 921 552807, Fax +49 921 552216
E-Mail: wi@uni-bayreuth.de
Internet: *h*ttp://wi.oec.uni-bayreuth.de

Evolution Strategies, Random Network Keys, and the One-Max Tree Problem

Barbara Schindler
Dept. of Information Systems
University of Bayreuth
Universitaetsstr. 30
D-95440 Bayreuth/Germany
barbara.schindler@stud.uni-bayreuth.de

Franz Rothlauf
Dept. of Information Systems
University of Bayreuth
Universitaetsstr. 30
D-95440 Bayreuth/Germany
rothlauf@uni-bayreuth.de

November 15, 2001

Abstract

Evolution strategies (ES) are efficient optimization methods for continuous problems. However, many combinatorial optimization methods can not be represented by using continuous representations. The development of the network random key representation (Rothlauf et al., 2000) which represents trees by using real numbers allows one to use ES for combinatorial tree problems.

In this paper we apply ES to tree problems using the network random key representation. We examine whether existing recommendations (Rechenberg, 1973a; Schwefel, 1995; Kursawe, 1999) regarding optimal parameter settings for ES, which were developed for the easy sphere and corridor model, are also valid for the easy one-max tree problem (Rothlauf et al., 2000).

The results show that the $\frac{1}{5}$-success rule for the (1+1)-ES results in low performance because the standard deviation is continuously reduced and we get early convergence. However, for the $(\mu + \lambda)$-ES and the (μ, λ)-ES the recommendations from the literature are confirmed for the parameters of mutation τ_1 and τ_2 and the ratio μ/λ. This paper illustrates how existing theory about ES is helpful in finding good parameter settings for new problems like the one-max tree problem.

1 Introduction

Evolution strategies (Schwefel, 1965; Rechenberg, 1965; Schwefel, 1968; Rechenberg, 1973b) are a class of direct, probabilistic search and optimization methods gleaned from the model of organic evolution. In contrast to genetic algorithms (GAs) (Holland, 1975), which work on binary strings and therefore process schemata, ES have been dedicated to continuous optimization problems. The main operator of ES is mutation, whereas recombination is only important for the self-adaption of the strategy parameters. Random network keys (NetKeys) have been proposed by Rothlauf, Goldberg, and Heinzl (2000) as a way to represent trees with continuous variables. This work was based on Bean (1992) and allows us to represent a permutation by a sequence of continuous variables.

In this work we want to investigate the performance of ES for tree problems when using the continuous NetKey representation. Because ES have been designed for solving continuous problems and have shown good performance therein, we expect ESs to perform well for network problems when using NetKeys. Furthermore, we want to examine whether the recommendations for the

setting of ES parameters, that are derived for the sphere and the corridor model, are also valid for the one-max tree problem. In analogy to the sphere and corridor models, the one-max tree problem (Rothlauf, Goldberg, & Heinzl, 2000; Rothlauf, 2001) is also fully easy and ES are expected to perform well. Finally, we compare the performance of ES and GAs for the one-max tree problem. We wanted to know which of the two search approaches, mutation versus crossover, performs better for this specific test problem.

The paper is structured as follows. In section 2 we present the NetKey encoding and present its major characteristics. This is followed by a short review of the one-max tree problem. In section 4, after taking a closer look at the different types of ES (subsection 4.1), we perform an analysis of the adjustment of ES parameters for the one-max problem (subsection 4.2), and finally compare the performance of ESs to GAs (subsection 4.3). The paper ends with concluding remarks.

2 Network Random Keys

This section gives a short overview about the NetKey encoding.

Network random keys (Rothlauf, Goldberg, & Heinzl, 2000) are adapted random keys (RKs) for the representation of trees. The RK encoding allows us to represent permutations and was first presented in Bean (1992). Originally, RKs were mainly used for the encoding of ordering and scheduling problems (Bean, 1994; Norman & Bean, 1994; Norman, 1995).

When using NetKeys, a key sequence of l random numbers $r_i \in [0,1]$, where $i \in \{0,\dots l-1\}$, represents a permutation r^s of length l. From the permutation r^s of length $l = n(n-1)/2$ a tree with n nodes and $n-1$ links is constructed using the following algorithm:

(1) Let $i = 0$, G be an empty graph with n nodes, and r^s the permutation of length $l = n(n-1)/2$ that can be constructed from the key sequence r. All possible links of G are numbered from 1 to l.

(2) Let j be the number at the ith position of the permutation r^s.

(3) If the insertion of the link with number j in G would not create a cycle, then insert the link with number j in G.

(4) Stop, if there are $n-1$ links in G.

(5) Increment i and continue with step 2.

With this calculation rule, we can construct a unique, valid tree from every possible key sequence. The use of the NetKey encoding has some important benefits:

- Standard crossover and mutation operators work properly. Therefore, the encoding has high locality and heritability.
- NetKeys allow a distinction between important and unimportant links.
- There is no over- or underspecification of a tree possible.

When applying mutation to an individual the offspring should be similar to the parent. Examining the NetKey encoding reveals that the mutation of one key results either in the same, or in a neighboring tree, which results in a high locality encoding. Furthermore, recombination operators should create offspring that inherit the properties of their parents. This means an offspring should inherit the links from its parents. Standard recombination operators, like x-point or uniform crossover, show this behavior when used for NetKeys: If a link exists in a parent, the value of the

corresponding key is high in comparison to the other keys. After recombination, the corresponding key in the offspring has the same, high value and is therefore also used with high probability for the construction of the tree.

A further benefit of the NetKey encoding is that genetic and evolutionary algorithms (GEAs) are able to distinguish between important and unimportant links. In contrast to other encodings, it is possible to identify the important links in the network. The algorithm which constructs a tree from the key sequence uses the high-quality links and ensures that they are not lost during the GEA run.

Finally, NetKeys always encode valid trees. No over- or underspecification is possible. The construction process which builds a tree from the key sequence ensures that NetKeys encode valid solutions only. Thus, we do not need an additional repair mechanism.

For a more detailed description of the NetKey encoding the reader is referred to Rothlauf et al. (2000) and Rothlauf, Goldberg, and Heinzl (2001).

3 The One-Max Tree Problem

This section gives a short overview of the one-max tree problem. For further information please refer to Rothlauf et al. (2000).

The one-max tree problem is based on the one-max problem for binary representations (Ackley, 1987). An optimal solution (tree) is chosen either randomly or by hand. The structure of this tree can be determined: It can be a star, a list, or an arbitrary tree with n nodes. In this work we only consider the optimal solution to be an arbitrary tree.

For the calculation of the fitness of the individuals, the distance d_{ab} between two trees G_a and G_b is used. It is defined as

$$d_{ab} = \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=0}^{i-1} |l_{ij}^a - l_{ij}^b|,$$

where l_{ij}^a is 1 if the link from node i to node j exists in tree G_a and 0 if it does not exist in G_a. n denotes the number of nodes. This definition of distance between two trees is based on the Hamming distance (Hamming, 1980) and $d_{ab} \in \{0, 1, \ldots, n-2\}$.

When using this distance metric for a minimization problem the fitness of an individual G_i is defined as the distance $d_{i,opt}$ to the optimal solution G_{opt}. Therefore, $f_i = d_{i,opt}$, and $f_i \in \{0, 1, \ldots, n-2\}$. An individual has fitness (cost) of $n-2$ if it has only one link in common with the best solution. If the two individuals do not differ ($G_i = G_{opt}$), the fitness (cost) of G_i is $f_i = 0$. In this work we only want to use a minimization problem. Because this test problem is similar to the standard one-max-problem it is easy to solve for mutation-based GEAs, but somewhat harder for recombination-based GAs (Goldberg, Deb, & Thierens, 1993).

4 Performance of Evolution Strategies and Adjustment of Parameters

In this section, after a short introduction into the functionality of evolution strategies, we present an investigation into the adjustment of ES parameters for the one-max tree problem when using the NetKey encoding. The section ends with a short comparison of ES and GA for this specific problem.

4.1 A Short Introduction into Evolution Strategies

ESs were developed by Ingo Rechenberg and Hans-Paul Schwefel in the 1960s at the Technical University of Berlin in Germany (Rechenberg, 1973a). First applications were experimental and dealt with hydrodynamical problems like shape optimization of a bended pipe, drag minimization of a joint plate (Rechenberg, 1965) and a structure optimization of a two-phase flashing nozzle (Schwefel, 1968).

The first (1 + 1)-ES used n-dimensional real valued vectors and created one offspring $x' = \{x'_1, \dots x'_n\}$ from one parent $x = \{x_1, \dots x_n\}$ by applying mutation with identical standard deviations σ to each parental allele x_i.

$$x'_i = x_i + \sigma \cdot N_i(0,1) \quad \forall i = 1, \dots, n.$$

$N(0,1)$ denotes a normal distributed one-dimensional random variable with expectation zero and standard deviation one. $N_i(0,1)$ indicates that the random variable is sampled anew for each possible value of the counter i. The resulting individual is evaluated and compared to its parent, and the better one survives to become the parent of the next generation. For the (1 + 1)-ES a theoretical convergence model for two specific problems, the sphere model and the corridor model, was developed. The $\frac{1}{5}$-success rule reflects the theoretical result that, to achieve fast convergence, on average one out of five mutations should result in higher fitness values. "The ratio of successful mutations to all mutations should be $\frac{1}{5}$. If it is greater than $\frac{1}{5}$, increase the standard deviation, if it is smaller, decrease the standard deviation." (Rechenberg, 1973a, p. 123).

To incorporate the principle of a population, Schwefel (1975) introduced the $(\mu + \lambda)$-ES and the (μ, λ)-ES. This notation considers the selection mechanism and the number of parents μ and offspring λ. For the $(\mu + \lambda)$-ES, the μ best individuals survive out of the union of the μ parents and the λ offspring. In the case of the (μ, λ)-ES, only the best μ offspring form the next parent generation. Both population-based ES start with a parent population of μ individuals. Each individual a consists of an n-dimensional vector $x \in \mathbb{R}^n$ and l standard deviations $\sigma \in \mathbb{R}_+$. One individual is described as $a = (x, \sigma)$ (Bäck, 1996).

For both, $(\mu + \lambda)$-ES and (μ, λ)-ES, recombination is used for the creation of the offspring. Mostly, discrete recombination is used for the decision variables x'_i and intermediate recombination is used for the standard deviations σ'_i. Discrete recombination means that x'_i is randomly taken from one parent, whereas intermediate recombination creates σ'_i as the arithmetic mean of the parents standard deviations.

However, the main operator in ES is mutation. It is applied to every individual after recombination:

$$\begin{aligned} \sigma'_k &= \sigma_k \cdot \exp(\tau_1 \cdot N(0,1) + \tau_2 \cdot N_k(0,1)) \quad \forall k = 1, 2, \dots, l, \\ x'_i &= x_i + \sigma'_i \cdot N_i(0,1) \qquad\qquad\qquad\quad \forall i = 1, 2, \dots, n. \end{aligned}$$

The standard deviations σ_k are mutated using a multiplicative, logarithmic normally distributed process with the factors τ_1 and τ_2. Then, the decision variables x_i are mutated by using the modified σ'_k. This mutation mechanism enables the ES to evolve its own strategy parameters during the search, exploiting an implicit link between appropriate internal model and good fitness values. One of the major advantages of ES is seen in its ability to incorporate the most important parameters of the strategy, e.g standard deviations, into the search process. Therefore, optimization not only takes place on object variables, but also on strategy parameters according to the actual local topology of the object function. This capability is called self-adaption.

4.2 Adjustment of Parameters

Over time, many recommendations for choosing ES parameters have been developed mainly for the simple sphere and the corridor model. We want to investigate if these recommendations also hold true for simple one-max tree problems represented using the real-valued NetKey encoding.

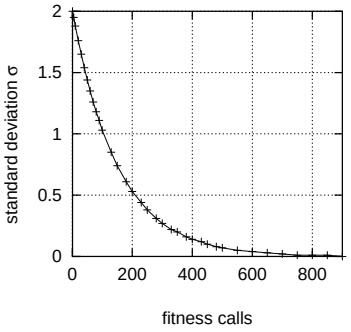


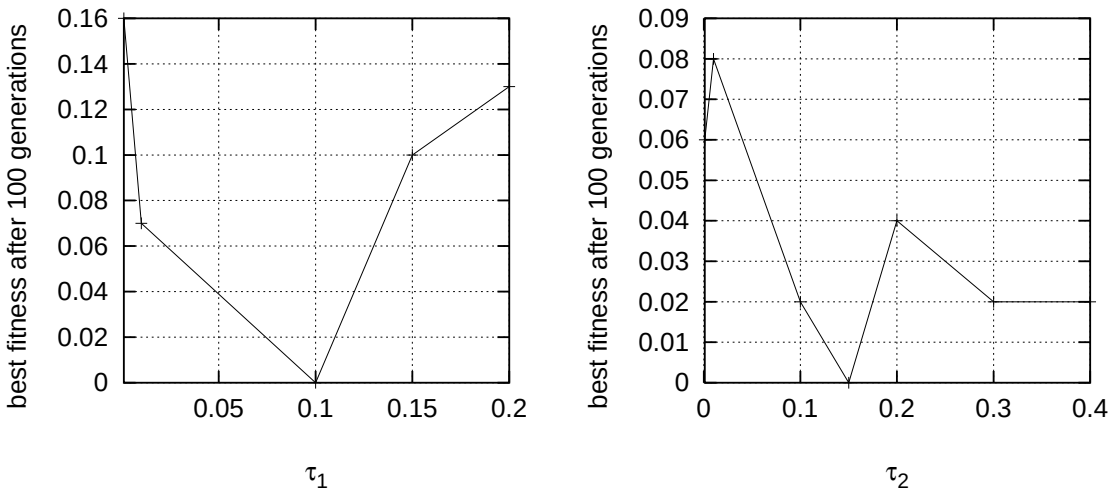
Figure 1: Standard deviation σ over number of fitness calls. We use the $\frac{1}{5}$-rule and a $(1+1)$-ES for the 8 node one-max tree problem.

When using the $(1+1)$-ES there are two possibilities for choosing the standard deviation σ. It can be either fixed to some value or adapted according to the $\frac{1}{5}$-success rule. For sphere and corridor models this rule results in fastest convergence. However, sometimes the probability of success cannot exceed $\frac{1}{5}$. For problems, where the objective function has discontinuous first partial derivatives, or at the edge of the allowed search space, the $\frac{1}{5}$-success rule does not work properly. Especially in the latter case, the success rule progressively forces the sequence of iteration points nearer to the boundary and the step lengths are continuously reduced without the optimum being approached with comparable accuracy (Schwefel, 1995).

Figure 3(a) and Figure 1 illustrate the problems of the $\frac{1}{5}$-success rule when using ES for solving an 8 node one-max tree problem. The plots show the fitness and the standard deviation σ over the number of fitness calls. The initial standard deviation $\sigma_0 = 2$ and we performed 200 runs. Due to the success rule the standard deviation is continuously reduced and we get early convergence. This behavior of $(1+1)$-ES can be explained when examining the NetKey encoding. In section 2 we saw that only $n-1$ out of $n(n-1)/2$ links are used for constructing the tree. Therefore, a mutation of one allele often does not result in a change of the represented tree. Only after on average about $1/n$ mutations does the structure of the tree change. However, for the $\frac{1}{5}$-rule we assume that every mutation results in a different phenotype and about every fifth new phenotype is superior to its parent. Therefore, the one-max tree problem is more difficult than the fully easy sphere and corridor models, and the $\frac{1}{5}$-rule can not be used.

Instead, we can use a fixed standard deviation σ. In Figure 3(b) we show the fitness after 10 000 iterations over the standard deviation σ for the 8 node one-max tree problem. The results show that the $(1+1)$-ES shows the best performance for a fixed standard deviation of $\sigma \approx 0.4$. Larger standard deviations do not result in a faster convergence but the search becomes random. With smaller standard deviations we also do not get better solutions, because with small standard deviations of mutation we only make slow progress.

To overcome the problem of the $(1+1)$-ES getting stuck in local optima, population-based ES approaches like $(\mu+\lambda)$-ES and (μ,λ)-ES have been proposed. We want to examine how one can adjust the strategy parameters τ_1 and τ_2. The standard deviations are mutated using a multiplicative, logarithmic normally distributed process. The logarithmic normal distribution is motivated as follows. A multiplicative modification process for the standard deviations guarantees positive values for σ and smaller modifications must occur more often than larger ones (Schwefel, 1977). Because the factors τ_1 and τ_2 are robust parameters, Schwefel (1977) suggests setting them as follows: $\tau_1 \propto (\sqrt{2\sqrt{n}})^{-1}$ and $\tau_2 \propto (\sqrt{2n})^{-1}$. Newer investigations indicate that optimal adjustments in the interval $[0.1, 0.2]$ (Kursawe, 1999; Nissen, 1997). τ_1 and τ_2 can be interpreted in the sense of "learning rates" as in artificial neural networks, and preliminary experiments with proportionality factors indicate that the search process can be tuned for particular objective functions by modifying these factors. We investigated in Figure 2 for the $(\mu+\lambda)$-ES whether the recommendations



(a) τ_2 fixed at 0.15

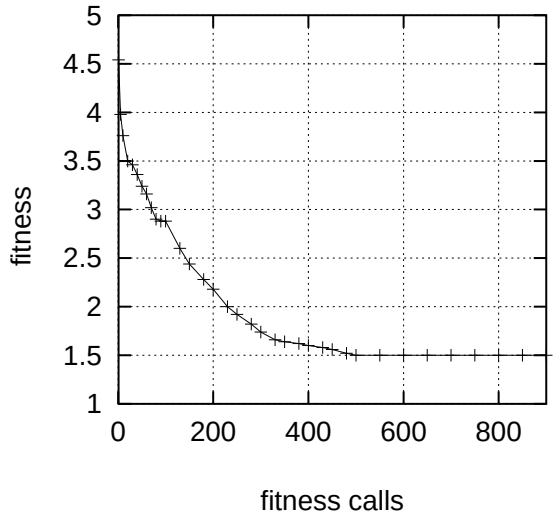
(b) τ_1 fixed at 0.1

Figure 2: Best fitness at end of the run over the strategy parameters τ_1 and τ_2 for the $(\mu + \lambda)$-ES. In Figure 2(a) we fixed τ_2 at 0.15 and varied τ_1, and in Figure 2(b) we fixed τ_1 at 0.1 and varied τ_2. We used $N = 200$, $\sigma_0 = 0.5$, $\mu/\lambda = 0.25$, 100 generations, and 1000 runs per plot.

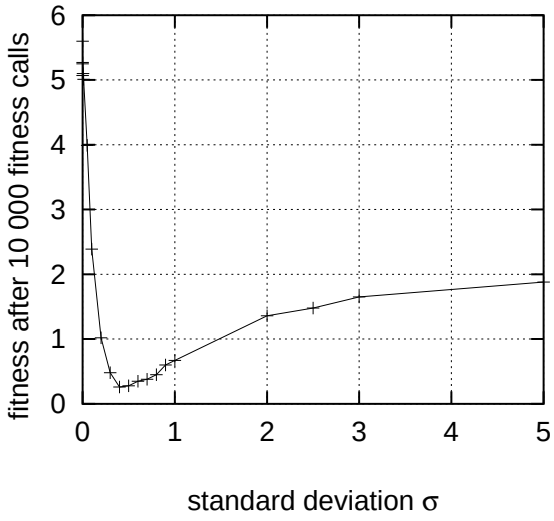
of Schwefel or Kursawe are also valid for the one-max tree problem. The plots show how the best fitness after 100 generations depends on τ_1 and τ_2. The results confirm the recommendations from Kursawe to initialize the parameters in the interval $[0.1, 0.2]$, where $\tau_2 > \tau_1$. The best solutions are $\tau_1 = 0.1$ and $\tau_2 = 0.15$ for the $(\mu + \lambda)$-ES and $\tau_1 = 0.15$ and $\tau_2 = 0.2$ for the (μ, λ)-ES.

The next part of our investigation focuses on the optimal proportion of μ parents to λ offspring to maximize the convergence velocity. Schwefel (1995) proposed a $(1, 5)$-ES or a $(1, 6)$-ES that is nearly optimal for sphere and corridor models. Figure 3(c) ($(\mu + \lambda)$-ES) and Figure 3(d) ((μ, λ)-ES) show the fitness over the number of fitness calls for the 8 node one-max tree problem. We used a population size of $N = \mu + \lambda = 200$, $\tau_1 = 0.13$, $\tau_2 = 0.16$, $\sigma_0 = 0.5$, and performed 1000 runs for every parameter setting. The results show that a ratio of $\frac{\mu}{\lambda} \in \{\frac{1}{4} \dots \frac{1}{7}\}$ results in good performance for the $(\mu + \lambda)$-ES and the (μ, λ)-ES. These results confirm the recommendations from Schwefel (1987) and Bäck (1996). The investigations for the sphere model indicated that the ratio of $\frac{\mu}{\lambda} \approx \frac{1}{7}$ is optimal concerning the accelerating effect of self-adaption. This ratio also provides the basic parameterization instrument for controlling the character of the search. Decreasing μ/λ emphasizes on path-oriented search and convergence velocity, while increasing μ/λ leads to a more volume-oriented search.

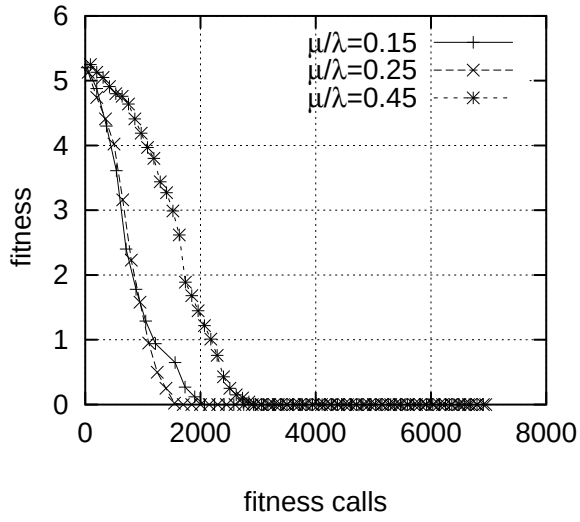
Finally, we want to examine the population size N which provides optimal convergence velocity. The population size mainly depends on the representation of the individuals and the optimization problem. Guidelines for choosing proper population sizes N when using NetKeys for the one-max tree problem and using selectorecombinative GAs were shown in Rothlauf, Goldberg, and Heinzl (2000). In Figure 3(e) we compare the fitness over the number of fitness calls for the $(\mu + \lambda)$-ES for different population sizes $N = \mu + \lambda$. We used $\tau_1 = 0.13$, $\tau_2 = 0.16$, $\sigma_0 = 0.5$, $\mu/\lambda = 0.25$ and performed 1000 runs for every parameter setting. The results reveal that for a 8 node problem, a population size $N = 200$ is enough to allow ES to find the optimal solution reliably and fast.



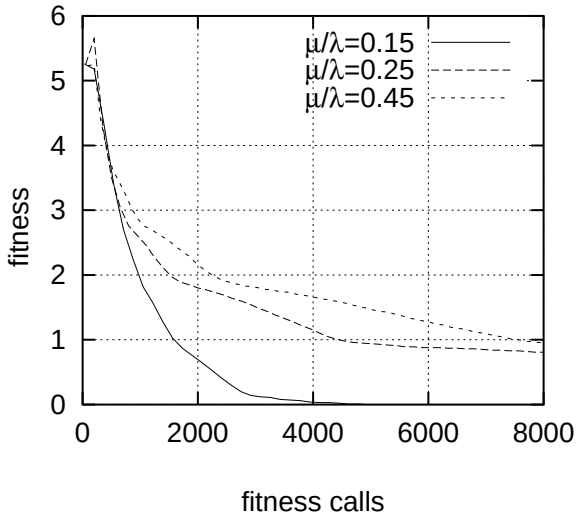
(a) Performance of $(1+1)$-ES using the $\frac{1}{5}$-success rule.



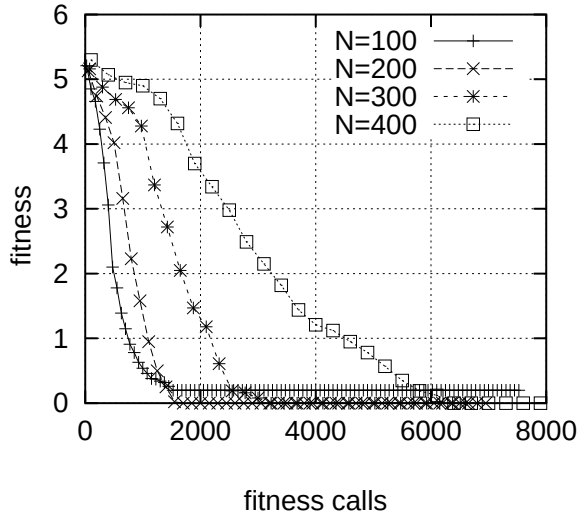
(b) $(1+1)$-ES using fixed σ (fitness after 10 000 fitness calls over σ).



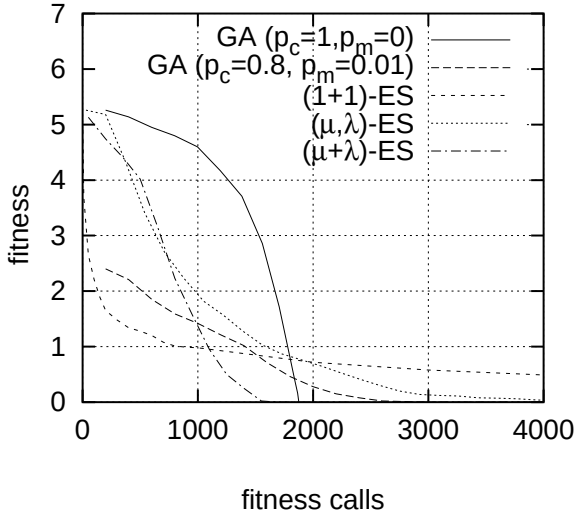
(c) $(\mu+\lambda)$-ES for different μ/λ ratios.



(d) (μ,λ)-ES for different μ/λ ratios.



(e) $(\mu+\lambda)$-ES for different $N=\mu+\sigma$.



(f) Comparison between ES and GA.

Figure 3: Adjustment of ES parameters for the 8 node one-max tree problem. All plots show the fitness over the number of fitness calls (except Figure 3(a) which shows best fitness after 10 000 fitness calls over σ.)

The results show that the simple $\frac{1}{5}$-rule for the $(1+1)$-ES from Rechenberg (1973a) does not work when using the NetKey representation. However, when using $(\mu+\lambda)$-ES or (μ,λ)-ES the recommendations for the simple sphere and corridor models from Schwefel (1977), Kursawe (1999), and Schwefel (1995) can also be used for the one-max tree problem using NetKeys. The existing guidelines help us to choose proper strategy parameters τ_1, τ_2, and the ratio $\frac{\mu}{\lambda}$. For further information about the use of ES for tree problems using the NetKey encoding the reader is referred to Schindler (2001).

4.3 A Comparison to Genetic Algorithms for the One-Max Tree Problem

After identifying optimal strategy parameters for ES we want to compare the performance of ES with GAs for the one-max tree problem using NetKeys.

For both optimization methods, ES and GA, we use uniform crossover. For the GA we implemented a roulette-wheel selection scheme and used $N = 200$. Mutation in the context of GA means that the value of one key is randomly changed. As before, we used for the ES $\tau_1 = 0.13$, $\tau_2 = 0.16$, $\sigma_0 = 0.5$, $N = \mu + \lambda = 200$, and performed 1000 runs for every parameter setting.

Figure 3(f) compares the performance of ES and GAs for the one-max tree problem. We plot the fitness over the number of fitness calls. The results show that a $(\mu+\lambda)$-ES has the highest performance. The $(1+1)$-ES gets stuck and is not able to find the optimal solution.

5 Conclusions

In this paper we extended the use of evolution strategies to combinatorial tree problems. Evolution strategies are designed for continuous optimization problems and can be applied to trees when using the continuous network random key (NetKey) representation. We examined for the simple one-max tree problem how to adjust the parameters of the $(1+1)$-, $(\mu+\lambda)$-, and (μ,λ)-ES and compared their performance to a simple GA.

The results showed that the recommendations regarding the adjustment of the ES parameters (τ_1, τ_2, and μ/λ) for simple sphere and corridor models (Rechenberg, 1973a; Schwefel, 1995; Kursawe, 1999) can also be used for the easy one-max tree problem when using the NetKey encoding. Only the $\frac{1}{5}$-success rule for the $(1+1)$-ES does not hold true for the one-max tree problem because most of the mutations do not change the represented tree. Therefore, the strategy parameter σ is continously reduced and the algorithm gets stuck.

The results indicate that existing theory about ES can often help in finding good parameter settings for new types of problems. We want to encourage researchers when developing new representations or techniques to first look at existing theory, to check if they can be used advantageously, and not to reinvent the wheel.

References

Ackley, D. H. (1987). *A connectionist machine for genetic hill climbing.* Boston: Kluwer Academic.

Bäck, T. (1996). *Evolutionary algorithms in theory and practice.* New York: Oxford University Press.

Bean, J. C. (1992, June). *Genetics and random keys for sequencing and optimization* (Technical Report 92-43). Ann Arbor, MI: Department of Industrial and Operations Engineering, University of Michigan.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, *6*(2), 154–160.

Goldberg, D. E., Deb, K., & Thierens, D. (1993). Toward a better understanding of mixing in genetic algorithms. *Journal of the Society of Instrument and Control Engineers*, *32*(1), 10–16.

Hamming, R. (1980). *Coding and information theory.* Prentice-Hall.

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Ann Arbor, MI: University of Michigan Press.

Kursawe, F. (1999). *Grundlegende empirische Untersuchungen der Parameter von Evolutionsstrategien - Metastrategien.* Doctoral dissertation, University of Dortmund.

Nissen, V. (1997). *Einführung in evolutionäre Algorithmen: Optimierung nach dem Vorbild der Evolution.* Wiesbaden: Vieweg.

Norman, B. A. (1995). *Scheduling using the random keys genetic algorithm.* unpublished PhD thesis, University of Michigan, Ann Arbor, Michigan.

Norman, B. A., & Bean, J. C. (1994). *Random keys genetic algorithm for job shop scheduling* (Tech. Rep. No. 94-5). Ann Arbor, MI: The University of Michigan.

Rechenberg, I. (1965). *Cybernetic solution path of an experimental problem* (Technical Report 1122). Farnborough, Hants., UK: Royal Aircraft Establishment, Library Translation.

Rechenberg, I. (1973a). Bionik, Evolution und Optimierung. *Naturwissenschaftliche Rundschau*, *26*(11), 465–472.

Rechenberg, I. (1973b). *Evolutionsstrategie Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart-Bad Cannstatt: Friedrich Frommann Verlag.

Rothlauf, F. (2001). *Towards a theory of representations for genetic and evolutionary algorithms: Development of basic concepts and their application to binary and tree representations.* Doctoral dissertation, University of Bayreuth/Germany.

Rothlauf, F., Goldberg, D. E., & Heinzl, A. (2000). *Network random keys – a tree network representation scheme for genetic and evolutionary algorithms* (Technical Report No. 8/2000). University of Bayreuth, Germany. to be published in Evolutionary Computation.

Rothlauf, F., Goldberg, D. E., & Heinzl, A. (2001, 7 July). On the debate concerning evolutionary search using Prüfer numbers. In Rothlauf, F. (Ed.), *Representations and Operators for Network Problems (ROPNET 2001)* (pp. 262–267). San Francisco, California, USA.

Schindler, B. (2001, Mai). *Einsatz von Evolutionären Stratgien zur Optimierung baumförmiger Kommunikationsnetzwerke.* Master's thesis, Universität Bayreuth, Lehrstuhl für Wirtschaftsinformatik.

Schwefel, H.-P. (1965). *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik.* Master's thesis, Technische Universität Berlin.

Schwefel, H.-P. (1968). *Experimentelle Optimierung einer Zweiphasendüse* (Bericht 35). AEG Forschungsinstitut Berlin, Projekt MHD-Staustahlrohr.

Schwefel, H.-P. (1975). *Evolutionsstrategie und numerische Optimierung.* Doctoral dissertation, Technical University of Berlin.

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie.* Basel: Birkhuser. from Interdisciplinary Systems Research, volume 26.

Schwefel, H.-P. (1987). Collective phenomena in evolutionary systems. In Checkland, P., & Kiss, I. (Eds.), *Problems of Constancy and Change - The Complementarity of Systems Approaches to Complexity*, Volume 2 (pp. 1025–1033). Budapest. Papers presented at the 31st Annual Meeting of the International Society for General System Research.

Schwefel, H.-P. (1995). *Evolution and optimum seeking*. New York: Wisley & Sons.