An Overview of Evolutionary Algorithms for Parameter Optimization

Thomas Bäck* Hans-Paul Schwefel†

University of Dortmund
Department of Computer Science · Chair of Systems Analysis
P.O. Box 50 05 00 · D–4600 Dortmund 50 · Germany

Abstract

Three main streams of *Evolutionary Algorithms* (EAs), i.e. probabilistic optimization algorithms based on the model of natural evolution, are compared with each other in this article: *Evolution Strategies* (ESs), *Evolutionary Programming* (EP), and *Genetic Algorithms* (GAs). The comparison is performed with respect to certain characteristic components of EAs, i.e. the representation scheme of object variables, mutation, recombination, and the selection operator. Furthermore, each algorithm is formulated in a high-level notation as an instance of the general, unifying basic algorithm, and the fundamental theoretical results on the algorithms are presented. Finally, after presenting experimental results for three test functions representing a unimodal and a multimodal case as well as a step function with discontinuities, similarities and differences of the algorithms are elaborated, and some hints to open research questions are sketched.

1 Introduction

Nearly three decades of research and applications have clearly demonstrated that the mimicked search process of natural evolution can yield very robust, direct computer algorithms, although these imitations are crude simplifications of biological reality. The resulting *Evolutionary Algorithms* are based on the collective learning process within a population of individuals, each of which represents a search point in the space of potential solutions to a given problem. The population is arbitrarily initialized, and it evolves towards better and better regions of the search space by means of randomized processes of *selection* (which is deterministic in some algorithms), *mutation*, and *recombination* (which is completely omitted in some algorithmic realizations). The environment (given aim of the search) delivers a quality information (*fitness value*) of the search points, and

*baeck@ls11.informatik.uni-dortmund.de
†schwefel@ls11.informatik.uni-dortmund.de

the selection process favours those individuals of higher fitness to reproduce more often than worse individuals. The recombination mechanism allows the mixing of parental information while passing it to their descendants, and mutation introduces innovation into the population.

This informal description is put into concrete terms by introducing some notational conventions. The notation forms the basis for the description of particular Evolutionary Algorithms and is not repeated in the following sections. Instead, in order to allow of a quick access the notation is summarized in a table at the end of the paper.

$f : \mathbb{R}^n \to \mathbb{R}$ denotes the objective function to be optimized, and without loss of generality we assume a minimization task in the following. In general, fitness and objective function values of an individual are not required to be identical, such that fitness $\Phi : I \to \mathbb{R}$ (I being the space of individuals) and f are distinguished mappings (but f is always a component of Φ). While $\vec{a} \in I$ is used to denote an individual, $\vec{x} \in \mathbb{R}^n$ indicates an object variable vector. Furthermore, $\mu \geq 1$ denotes the size of the parent population and $\lambda \geq 1$ is the offspring population size, i.e. the number of individuals created by means of recombination and mutation at each generation (if the life time of individuals is limited to one generation, i.e. selection is not elitist, $\lambda > \mu$ is a more reasonable assumption). A population at generation t, $P(t) = \{\vec{a}_1(t), \ldots, \vec{a}_\mu(t)\}$, consists of individuals $\vec{a}_i(t) \in I$, and $r_{\Theta_r} : I^\mu \to I^\lambda$ denotes the recombination operator which might be controlled by additional parameters summarized in the set Θ_r. Similarly, the mutation operator $m_{\Theta_m} : I^\lambda \to I^\lambda$ modifies the offspring population, again being controlled by some parameters Θ_m. Both mutation and recombination, though introduced here as macro-operators transforming populations into populations, can essentially be reduced to local operators $m'_{\Theta_m} : I \to I$ and $r'_{\Theta_r} : I^\mu \to I$ that create one individual when applied. These local variants will be used when the particular instances of the operators are explained in subsequent sections. Selection $s_{\Theta_s} : (I^\lambda \cup I^{\mu+\lambda}) \to I^\mu$ is applied to choose the parent population of the next generation. During the evaluation step the fitness function $\Phi : I \to \mathbb{R}$ is calculated for all individuals of a population, and $\iota : I^\mu \to \{\text{true}, \text{false}\}$ is used to denote the termination criterion. The resulting algorithmic description is given below.

ALGORITHM 1 (Outline of an Evolutionary Algorithm)

> $t := 0;$
> *initialize* $P(0) := \{\vec{a}_1(0), \ldots, \vec{a}_\mu(0)\} \in I^\mu;$
> *evaluate* $P(0) : \{\Phi(\vec{a}_1(0)), \ldots, \Phi(\vec{a}_\mu(0))\};$
> **while** $(\iota(P(t)) \neq$ **true**$)$ **do**
>   *recombine:* $P'(t) := r_{\Theta_r}(P(t));$
>   *mutate:* $P''(t) := m_{\Theta_m}(P'(t));$
>   *evaluate* $P''(t) : \{\Phi(\vec{a}''_1(t)), \ldots, \Phi(\vec{a}''_\lambda(t))\};$
>   *select:* $P(t+1) := s_{\Theta_s}(P''(t) \cup Q);$
> $t := t + 1;$
> **od**

Here, $Q \in \{\emptyset, P(t)\}$ is a set of individuals that are additionally taken into account during the selection step. The evaluation process yields a multiset of fitness values, which in general are not necessarily identical to objective function values. However, since the selection criterion operates on fitness instead of objective function values, fitness values are used here as result of the evaluation process. During the calculation of fitness, the evaluation of objective function values is always necessary, such that the information is available and can easily be stored in an appropriate data structure.

Three main streams of instances of this general algorithm, developed independently of each other, can nowadays be identified: *Evolutionary Programming* (EP), originally developed by L. J. Fogel, A. J. Owens, and M. J. Walsh in the U.S. (Fogel et al. 1966) and recently refined by D. B. Fogel (Fogel 1991), *Evolution Strategies* (ESs), developed in Germany by I. Rechenberg (Rechenberg 1965; Rechenberg 1973) and H.-P. Schwefel (Schwefel 1977; Schwefel 1981), and *Genetic Algorithms* (GAs) by J. Holland (Holland 1975) in the U.S. as well, with refinements by K. De Jong (De Jong 1975), J. Grefenstette (Grefenstette 1986; Grefenstette 1987b), and D. Goldberg (Goldberg 1989). Each of these main stream algorithms have clearly demonstrated their capability to yield good approximate solutions even in case of complicated multimodal, discontinuous, non-differentiable, and even noisy or moving response surfaces of optimization problems. A variety of applications has been presented in the conference proceedings (Grefenstette 1985; Grefenstette 1987; Schaffer 1989; Belew and Booker 1991) (GAs), (Fogel and Atmar 1992) (EP), and (Schwefel and Männer 1991; Männer and Manderick 1992) (GAs & ESs as well as other natural metaphors), and an annotated bibliography collected at the University of Dortmund actually contains 260 references to applications of EAs (Bäck et al. 1992).

Within each research community, parameter optimization has been a common and highly successful theme of applications. Evolutionary Programming and especially Genetic Algorithms, however, were designed with a very much broader range of possible applications in mind and confirmed their wide applicability by a variety of important examples in fields like machine learning, control, automatic programming, planning, and others. A comparison of these algorithms as performed here can therefore never be complete with respect to the application range but must necessarily focus on the common denominator, the parameter optimization problem. This is done here both to formulate and formalize a general description of the algorithms in their parameter optimization instance as well as to provide a first step towards more detailed empirical and theoretical comparison.

Since these algorithms bear so many similarities due to their reliance on organic evolution, it is a surprising fact that general comparisons were not performed earlier. It is a central hope of the authors that this article may provide an appropriate overview of the strong similarities of the three main stream algorithms in their parameter optimization form and thus stimulates further discussions between the so far almost isolated research groups.

2 Comparison

To simplify the comparison of the algorithms, some categories of information guided along the algorithm's outline (1) are presented for each of the three algorithms. Each subsection is introduced by a short historical overview of the algorithm and then turns to describing the fitness evaluation and representation of search points, mutation, recombination, and selection mechanisms. Furthermore, the algorithms are presented in an abstract, high-level notation emphasizing specific concepts of and differences between the algorithms, and a short overview of the basic theory is given for each of the algorithms. Because of their similar representation of search points, we start with discussing ESs and EP, then turning to GAs. Since the focus is on parameter optimization, search points — whatever way of representing them may be used by the particular algorithm — correspond to n-dimensional real-valued vectors $\vec{x} \in \mathbb{R}^n$.

In order to simplify the explanation of the algorithms, the following mathematical notations will be used: $N(0,1)$ denotes a realization of a normally distributed one-dimensional random variable having expectation zero and standard deviation 1, while $N_i(0,1)$ indicates that the random variable is sampled anew for each value of the counter i. A realization of a normally distributed one-dimensional random variable having expectation zero and standard deviation σ is then given by $\sigma \cdot N(0,1)$.

2.1 Evolution Strategies

First efforts towards an ES took place in 1964 at the Technical University of Berlin (TUB) in Germany. Applications were mainly experimental and dealt with hydrodynamical problems like shape optimization of a bended pipe and of a flashing nozzle. Different versions of the strategy were simulated also on the first available computer at TUB (Schwefel 1965). Rechenberg developed a theory of convergence velocity for the so-called (1+1)–ES, a simple mutation-selection mechanism working on one individual which creates one offspring per generation by means of Gaussian mutation, and he proposed a theoretically confirmed rule for changing the standard deviation of mutations exogenously (1/5-*success rule*) (Rechenberg 1973) (see also section 2.1.6). He also proposed a first *multimembered* ES, a $(\mu{+}1)$–ES where $\mu \geq 1$ individuals recombine to form one offspring, which after mutation eventually replaces the worst parent individual in a similar way as within the Simplex method. This strategy, though never widely used, provided the basic idea to facilitate the transition to $(\mu{+}\lambda)$–ES and (μ,λ)–ES introduced and investigated by Schwefel (Schwefel 1977; Schwefel 1981), the resulting strategies (especially the latter one) being state-of-the-art in ES research. In their most general form, these strategies are described here.

2.1.1 Fitness Evaluation and Representation

Search points in ESs are n-dimensional vectors $\vec{x} \in \mathbb{R}^n$, and the fitness value of an individual is identical to its objective function value, i.e. $\Phi(\vec{a}) = f(\vec{x})$ where $\vec{x}$ is the object variable component of $\vec{a}$. Additionally, each individual may include up to n different variances $c_{ii} = \sigma_i^2$ ($i \in \{1,\ldots,n\}$), as well as up to $n \cdot (n-1)/2$ covariances c_{ij} ($i \in \{1,\ldots,n-1\}$, $j \in \{i+1,\ldots,n\}$) of the generalized n-dimensional normal distribution with expectation vector $\vec{0}$, having probability density function

$$p(\vec{z}) = \sqrt{\frac{\det \mathbf{A}}{(2\pi)^n}} \exp\left(-\frac{1}{2}\vec{z}^{\,T}\mathbf{A}\vec{z}\right) \quad , \tag{1}$$

where $\mathbf{A}^{-1} = (c_{ij})$ represents the covariance matrix, and $\vec{z}$ denotes the random variable. Altogether, up to $w = n \cdot (n+1)/2$ strategy parameters can be combined with object variables to form an individual $\vec{a} \in I = \mathbb{R}^{n+w}$. Often, however, only the variances are taken into account, such that $\vec{a} \in I = \mathbb{R}^{2n}$, and it is sometimes even useful to work with one common variance valid for all object variables, i.e. $\vec{a} \in I = \mathbb{R}^{n+1}$. To assure positive-definiteness of the covariance matrix $\mathbf{A}^{-1}$, the algorithm uses the equivalent rotation angles α_{ij} ($\tan 2\alpha_{ij} = 2c_{ij}/(\sigma_i^2 - \sigma_j^2)$). Furthermore, (mean) mutation step sizes σ_i, i.e. the standard deviations, are used in the implementation rather than the variances σ_i^2. In the following, we use the notation $\vec{N}(\vec{0}, \vec{\sigma}, \vec{\alpha})$ to denote a realization of a random vector distributed according to the generalized n-dimensional normal distribution having expectation $\vec{0}$, standard deviations $\vec{\sigma}$, and rotation angles $\vec{\alpha}$. $\vec{a} = (\vec{x}, \vec{\sigma}, \vec{\alpha}) \in \mathbb{R}^{n+w}$ is used to denote a complete individual.

2.1.2 Mutation

In its most general form the (local) mutation operator $m'_{\{\tau,\tau',\beta\}} : I \to I$ (where $I = \mathbb{R}^{n+w}$) yields a mutated individual $m'_{\{\tau,\tau',\beta\}}(\vec{a}) = (\vec{x}', \vec{\sigma}', \vec{\alpha}')$ by first mutating the standard deviations and rotation angles and then mutating the object variables according to the now modified probability density function of individual $\vec{a}$, i.e. $\forall i \in \{1, \ldots, n\}$, $\forall j \in \{1, \ldots, n \cdot (n-1)/2\}$:

$$\begin{aligned} \sigma_i' &= \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1)) \\ \alpha_j' &= \alpha_j + \beta \cdot N_j(0,1) \\ \vec{x}' &= \vec{x} + \vec{N}(\vec{0}, \vec{\sigma}', \vec{\alpha}') \quad . \end{aligned} \tag{2}$$

Mutations of the object variables now may be linearly *correlated* according to the values of $\vec{\alpha}$, and $\vec{\sigma}$ provides a scaling of the metrics. The global factor $\tau' \cdot N(0,1)$ allows for an overall change of the mutability, whereas $\tau \cdot N_i(0,1)$ allows for individual changes of the "mean step sizes" σ_i. The factors τ, τ' and β are rather robust exogenous parameters, which Schwefel suggests to set as follows:

$$\begin{aligned} \tau &\propto \left(\sqrt{2\sqrt{n}}\right)^{-1} \\ \tau' &\propto \left(\sqrt{2n}\right)^{-1} \\ \beta &\approx 0.0873 \quad . \end{aligned} \tag{3}$$

Usually, the proportionality constants for τ and τ' have the value one, and the value suggested for β (in radians) equals $5°$. This special mutation mechanism enables the algorithm to evolve its own strategy parameters (standard deviations and covariances) during the search, exploiting an implicit link between appropriate internal model and good fitness values. The resulting evolution and adaptation of strategy parameters according to the topological requirements has been termed *self-adaptation* by Schwefel (Schwefel 1987).

2.1.3 Recombination

Different recombination mechanisms are used in ESs either in their usual form, producing one new individual from two randomly selected parent individuals, or in their *global* form, allowing of taking components for one new individual from potentially all individuals available in the parent population. Furthermore, recombination is performed on strategy parameters as well as on the object variables, and the recombination operator may be different for object variables, standard deviations, and rotation angles. Recombination rules for an operator $r' : I^\mu \rightarrow I$ creating an individual $r'(P(t)) = \vec{a}' = (\vec{x}', \vec{\sigma}', \vec{\alpha}') \in I$ are given here representatively only for the object variables ($\forall i \in \{1, \ldots, n\}$):

$$x'_i = \begin{cases} x_{S,i} & \text{without recombination} \\ x_{S,i} \text{ or } x_{T,i} & \text{discrete recombination} \\ x_{S,i} + \chi \cdot (x_{T,i} - x_{S,i}) & \text{intermediate recombination} \\ x_{S_i,i} \text{ or } x_{T_i,i} & \text{global, discrete} \\ x_{S_i,i} + \chi_i \cdot (x_{T_i,i} - x_{S_i,i}) & \text{global, intermediate} \end{cases} \tag{4}$$

Indices S and T denote two parent individuals selected at random from $P(t)$, and $\chi \in [0,1]$ is a uniform random variable. For the global variants, for each component of $\vec{x}$ the parents S_i, T_i as well as χ_i are determined anew. Empirically, discrete recombination on object variables and intermediate recombination on strategy parameters have been observed to give best results. Recombination of strategy parameters has shown to be mandatory for this mechanism to work. Historically, intermediate recombination and its global form have always been used with a fixed value of $\chi = 1/2$ and only recently Schwefel proposed the generalization indicated in (4).

2.1.4 Selection

Selection in ESs is completely deterministic, selecting the μ best ($1 \leq \mu < \lambda$) individuals out of the set of λ offspring individuals ((μ,λ)-selection) or out of the union of parents and offspring ((μ+λ)-selection). Although the (μ+λ)-selection is elitist and therefore guarantees a monotonously improving performance, this selection strategy is unable to deal with changing environments and jeopardizes the self-adaptation mechanism with respect to the strategy parameters (internal model), especially within small populations. Therefore, the (μ,λ)-selection is recommended nowadays, investigations indicating a ratio for $\mu/\lambda \approx 1/7$ being optimal (Schwefel 1987).

2.1.5 Conceptual Algorithm

Combining the previous topics, the following conceptual algorithm of a (μ+λ)–ES and (μ,λ)–ES, respectively, results:

Algorithm 2 (Outline of an ES)

$t := 0;$
initialize $P(0) := \{\vec{a}_1(0), \ldots, \vec{a}_\mu(0)\} \in I^\mu$
  where $I = I\!R^{n+w}$
  and $\vec{a}_k = (x_i,\ \sigma_i,\ \alpha_j\ \forall i \in \{1,\ldots,n\}\ ,\ \forall j \in \{1,\ldots,n\cdot(n-1)/2\});$
evaluate $P(0):\ \{\Phi(\vec{a}_1(0)), \ldots, \Phi(\vec{a}_\mu(0))\}$
  where $\Phi(\vec{a}_k(0)) = f(\vec{x}_k(0));$
while $(\iota(P(t)) \neq$ **true**) **do** % while termination criterion not fulfilled
  recombine: $\vec{a}'_k(t) := r'(P(t))\ \forall k \in \{1,\ldots,\lambda\};$
  mutate: $\vec{a}''_k(t) := m'_{\{\tau,\tau',\beta\}}(\vec{a}'_k(t))\ \forall k \in \{1,\ldots,\lambda\};$
  evaluate: $P''(t) := \{\vec{a}''_1(t), \ldots, \vec{a}''_\lambda(t)\}$:
    $\{\Phi(\vec{a}''_1(t)), \ldots, \Phi(\vec{a}''_\lambda(t))\}$ **where** $\Phi(\vec{a}''_k(t)) = f(\vec{x}''_k(t));$
  select: $P(t+1) :=$ **if** *(μ,λ)-selection*
    then $s_{(\mu,\lambda)}(P''(t));$
    else $s_{(\mu+\lambda)}(P(t) \cup P''(t));$
  $t := t+1;$
od

2.1.6 Theory

The 1/5 success rule for a (1+1)–ES was derived by Rechenberg by investigating the model functions

$$f_1(\vec{x}) = c_0 + c_1 x_1 \tag{5}$$

(*corridor model*), where $\forall i \in \{2,\ldots,n\} : -b/2 \leq x_i \leq b/2$, and

$$f_2(\vec{x}) = c_0 + c_1 \sum_{i=1}^{n} (x_i - x_i^*)^2 = c_0 + c_1 r^2 \tag{6}$$

(*sphere model*), where $\vec{x}^*$ denotes the minimum and r is the actual Euclidean distance between $\vec{x}$ and $\vec{x}^*$. c_0 and $c_1 \neq 0$ denote arbitrary constants. For both functions, Rechenberg calculated the expectation values φ_1, φ_2 for the rates of convergence, expressed here in terms of dimensionless, normalized variables $\sigma'_1 = \sigma_1 n/b$, $\varphi'_1 = \varphi_1 n/b$, $\sigma'_2 = \sigma_2 n/r$, and $\varphi'_2 = \varphi_2 n/r$:

$$\begin{aligned} \varphi'_1 &\approx \frac{\sigma'_1}{\sqrt{2\pi}} \exp\left(-\sqrt{\frac{2}{\pi}}\sigma'_1\right) && \text{for } n \gg 1\ , \\ \varphi'_2 &= \frac{\sigma'_2}{\sqrt{2\pi}} \exp\left(-\frac{\sigma'^2_2}{8}\right) - \frac{\sigma'^2_2}{4}\left(1 - \operatorname{erf}\left(\frac{\sigma'_2}{\sqrt{8}}\right)\right) && \text{for } n \gg 1\ , \end{aligned} \tag{7}$$

where erf denotes the error function. Based on these expressions, it is possible to determine the optimal standard deviations σ'^*_i by setting $\left.\frac{d\varphi'_i}{d\sigma'_i}\right|_{\sigma'^*_i} = 0$ and to calculate the maximum convergence rates $\varphi'^*_i = \varphi'_i(\sigma'^*_i)$:

$$\begin{aligned} \sigma'^*_1 &= \sqrt{\tfrac{\pi}{2}} \approx 1.253 & \sigma'^*_2 &\approx 1.224 \\ \varphi'^*_1 &= \tfrac{1}{2e} \approx 0.184 & \varphi'^*_2 &\approx 0.2025 \end{aligned} \tag{8}$$

Furthermore, expectation values of the probabilities $p_i = \mathcal{P}\{f_i(m'(\vec{x})) \leq f_i(\vec{x})\}$ for a successful mutation were also calculated for both functions:

$$\begin{aligned} p_1 &\approx \frac{1}{2}\exp\left(-\sqrt{\frac{2}{\pi}}\sigma_1'\right) && \text{for } n \gg 1 \ , \\ p_2 &= \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{\sigma_2'}{\sqrt{8}}\right)\right) && \text{for } n \gg 1 \ , \end{aligned} \tag{9}$$

thus allowing to calculate the optimal success probabilities $p_i^* = p_i(\sigma_i'^*)$:

$$p_1^* \ = \ \tfrac{1}{2e} \ \approx \ 0.184 \qquad p_2^* \ \approx \ 0.270 \ . \tag{10}$$

Both are close to the value 1/5, and based on this result Rechenberg formulated the 1/5 success rule as a method for controlling the standard deviation during the optimization process according to the measured success frequency ((Rechenberg 1973), p. 122):

> The ratio of successful mutations to all mutations should be 1/5. If it is greater than 1/5, increase the standard deviation, if it is smaller, decrease the standard deviation.

Later on, Schwefel suggested to measure the ratio over $10n$ trials and to use a multiplicative factor $0.82 \leq c \leq 1$ for decreasing and $1/c$ for increasing the standard deviation (Schwefel 1981).

There are two problems, however, with respect to using the 1/5-success rule. The first one arises when the topological situation does not lead to success rates larger than 1/5 by decreasing the mutation step size towards zero. This often happens along active constraints, or in case of corners of level sets $f = \text{const}$ if f is not continuously differentiable or even discontinuous. Secondly, the 1/5 success rule gives no hint to handling σ_i individually and therefore does not enable a suitable scaling of mutation step sizes along different axes of the coordinate system.

Therefore, these results are useful for an algorithm with one standard deviation only, which does not use a real population, recombination, or self-adaptation, theoretical treatment of which is rather complicated. An extension to describe the expected progress rate of the population average for a (μ+λ)–ES and a (μ,λ)–ES using one single standard deviation and without recombination or self-adaptation was performed by Schwefel, resulting in the general convergence rate expression (Schwefel 1981)

$$\varphi \ = \ \frac{\lambda}{\mu}\int_{k'=k_{\min}}^{\infty} k' \cdot \sum_{i=1}^{\mu} \binom{\lambda-1}{i-1} p_{k_j=k'} \cdot p_{k_j<k'}^{\lambda-i} \cdot (1 - p_{k_j<k'})^{i-1} \, dk' \ , \tag{11}$$

where $k_{\min} = 0$ for a (μ+λ)–ES, $k_{\min} = -\infty$ for a (μ,λ)–ES, and $p_{k_j=k'}$ ($p_{k_j<k'}$) denotes the probability that a certain offspring individual j covers exactly the improvement distance k' (a smaller one). Since the outermost integration can not be performed even for a (1,λ)-ES analytically, Schwefel had to rely on additional assumptions and finally arrived at values $\lambda_1^* = 6$ and $\lambda_2^* = 4.7 \approx 5$ that maximize the functions φ_i^*/λ_i and therefore

yield the most reasonable compromise between computational effort (number of offspring individuals) and progress rate in the case of sequential processing.

This section is concluded by mentioning the most recent proof of global convergence with probability one for the (1+1)–ES, formulated by Rudolph (Rudolph 1992). Assuming p_t to denote the probability that the algorithm reaches the level set $L_{f^*+\varepsilon}$ in step t, $f > -\infty$, $\sum_{t=0}^{\infty} p_t = \infty$, $\mu(L_{f^*+\varepsilon}) > 0$ (Lebesgue measure), and a continuous density of the probability distribution used for mutations, he shows $\mathcal{P}\{\lim_{t\to\infty} \vec{x}(t) \in L_{f^*+\varepsilon}\} = 1$. This proof may be extended to a (μ+λ)–ES, but not for a (μ,λ)–ES. Though such a result is of no practical worth, it demonstrates that the method fulfills the minimum demands made on global optimization algorithms.

2.2 Evolutionary Programming

In the mid-sixties, L. J. Fogel et al. described EP for the evolution of finite state machines in order to solve prediction tasks (Fogel et al. 1966). The state transition tables of these machines were modified by uniform random mutations on the corresponding discrete, finite alphabet. Evaluation of fitness took place according to the number of symbols predicted correctly. Each machine in the parent population generated one offspring by means of mutation, and the best half number of parents and offspring were selected to survive, which would be called a (μ+μ)-strategy in ES-terminology. EP has been extended by D. B. Fogel to work on real-valued object variables based on normally distributed mutations, and the following description is based on (Fogel 1992a; Fogel 1992b).

2.2.1 Fitness Evaluation and Representation

For initialization, EP assumes a bounded subspace $\prod_{i=1}^{n}[u_i, v_i] \subset \mathbb{R}^n$ with $u_i < v_i$. Afterwards, however, the search domain is extended to $I = \mathbb{R}^n$, i.e. individuals are object variable vectors $\vec{a} = \vec{x} \in I$. In (Fogel 1992b), a concept of *meta-evolutionary programming* is presented which is rather similar to self-adaptation of standard deviations in ESs. To incorporate a vector $\vec{\nu} \in \mathbb{R}^{+^n}$ of variances (again initialized in a restricted domain $[0, c]^n$, $c > 0$ being an exogenous constant) rather than standard deviations ($\nu_i = \sigma_i^2$), the space of individuals is extended to $I' = \mathbb{R}^n \times \mathbb{R}^{+^n}$. Since EP is probably not known very widely, we will discuss both approaches in the following and therefore use the notation I for the standard EP individual space and I' for the space in meta-EP. The fitness values $\Phi(\vec{a})$ are obtained from objective function values $f(\vec{x})$ by scaling them to positive values and possibly by imposing some random alteration κ: $\Phi(\vec{a}) = \delta(f(\vec{x}), \kappa)$, where δ denotes the scaling function. To meet the requirement of a useful scaling, it is of the general form $\delta : \mathbb{R} \times S \to \mathbb{R}^+$ where S is an additional set of parameters involved in the process.

2.2.2 Mutation

In case of a standard EP, the Gaussian mutation operator $m'_{\{\beta_1,\ldots,\beta_n,\gamma_1,\ldots,\gamma_n\}} : I \to I$, $m'_{\{\beta_1,\ldots,\beta_n,\gamma_1,\ldots,\gamma_n\}}(\vec{x}) = \vec{x}'$, uses a standard deviation which is obtained for each component x_i as the square root of a linear transformation of the fitness value $\Phi(\vec{a}) = \Phi(\vec{x})$, i.e. $(\forall i \in \{1,\ldots,n\})$:

$$\begin{aligned} x_i' &= x_i + \sigma_i \cdot N_i(0,1) \\ \sigma_i &= \sqrt{\beta_i \cdot \Phi(\vec{x}) + \gamma_i} \quad . \end{aligned} \tag{12}$$

The proportionality constants β_i and the offsets γ_i together are $2n$ exogenous parameters which must be tuned for a particular task. Often, however, β_i and γ_i are set to one and zero, respectively, such that $x_i' = x_i + \sqrt{\Phi(\vec{x})} \cdot N_i(0,1)$.

To overcome the tuning difficulties with this approach, meta-EP self-adapts n variances per individual quite similar to ESs. Mutation $m'_{\{\alpha\}} : I' \to I'$, $m'_{\{\alpha\}}(\vec{a}) = (\vec{x}', \vec{\nu}')$ works as follows $(\forall i \in \{1,\ldots,n\})$:

$$\begin{aligned} x_i' &= x_i + \sqrt{\nu_i} \cdot N_i(0,1) \\ \nu_i' &= \nu_i + \sqrt{\alpha\nu_i} \cdot N_i(0,1) \quad . \end{aligned} \tag{13}$$

Here, α denotes an exogenous parameter assuring that ν_i tends to remain positive. Whenever by means of mutation a variance would become negative or zero, it is set to a small value $\varepsilon > 0$. Although the idea is the same as in ESs, the underlying stochastic process is fundamentally different. The log-normally distributed alterations in ESs automatically guarantee positiveness of σ_i, as well as no drift in case of zero selection pressure, whereas fluctuations in the meta-EP algorithm are expected to be much larger than in the ES and are not neutral. This mechanism surely deserves further investigation.

2.2.3 Recombination

A recombination operator combining features of different individuals occurring in the population similar to the recombination operator of ESs or GAs is *not* used within EP. In EP, however, each complete solution is generally viewed as a reproducing "population", such that the operator called "mutation" simulates all changes that transpire between one generation and the next (D. Fogel, personal communication 1992). This includes mutations and errors in transcription during sexual recombination, but raises a conceptual difference in the level of abstraction from the biological model between EP and ESs, GAs.

2.2.4 Selection

After creating μ offspring from μ parent individuals by mutating each parent once, a variant of stochastic q-tournament selection ($q \geq 1$ being a parameter of the algorithm) selects μ individuals from the union of parents and offspring, i.e. a randomized $(\mu+\mu)$-selection is used. In principle, for each individual $\vec{a}_k \in P(t) \cup P'(t)$, where $P'(t)$ is the population of mutated individuals, q individuals are chosen at random from $P(t) \cup P'(t)$ and compared to $\vec{a}_k$ with respect to their fitness values. Then, for $\vec{a}_k$, one counts how many of the q individuals are worse than $\vec{a}_k$, resulting in a score $w_k \in \{0,\ldots,q\}$. After doing so for all 2μ individuals, the individuals are ranked in descending order of the score values w_i $(i \in \{1,\ldots,2\mu\})$, and the μ individuals having highest score w_i are selected to form the next population. More formally, we have ($\forall i \in \{1, \ldots, 2\mu\}$):

$$w_i = \sum_{j=1}^{q} \begin{cases} 1 & \text{if } \Phi(\vec{a}_i) \leq \Phi(\vec{a}_{\chi_j}) \\ 0 & \text{otherwise} \end{cases} . \tag{14}$$

$\chi_j \in \{1, \ldots, 2\mu\}$ is a uniform integer random variable, sampled anew for each comparison. As the tournament size q is increased, the mechanism more and more becomes a deterministic $(\mu+\mu)$-scheme. Since the best individual is assigned a guaranteed maximum fitness score q, survival of the best is guaranteed (this property of an EA is usually called *elitist*).

2.2.5 Conceptual Algorithm

In the framework of our general algorithm outline, the following formulation for a meta-EP algorithm is derived:

ALGORITHM 3 (Outline of an EP algorithm)

> $t := 0;$
> *initialize* $P(0) := \{\vec{a}_1(0), \ldots, \vec{a}_\mu(0)\} \in I^\mu$
>   **where** $I = \mathbb{R}^n \times \mathbb{R}^{+n}$
>   **and** $\vec{a}_k = (x_i,\ \nu_i,\ \forall i \in \{1, \ldots, n\});$
> *evaluate* $P(0) :\ \{\Phi(\vec{a}_1(0)), \ldots, \Phi(\vec{a}_\mu(0))\}$
>   **where** $\Phi(\vec{a}_k(0)) = \delta(f(\vec{x}_k(0)), \kappa_k);$
> **while** $(\iota(P(t)) \neq$ **true**$)$ **do**   % while termination criterion not fulfilled
>   *mutate:* $\vec{a}'_k(t) := m'_{\{\alpha\}}(\vec{a}_k(t))\ \forall k \in \{1, \ldots, \mu\};$
>   *evaluate:* $P'(t) := \{\vec{a}'_1(t), \ldots, \vec{a}'_\mu(t)\} :$
> $\{\Phi(\vec{a}'_1(t)), \ldots, \Phi(\vec{a}'_\mu(t))\}$ **where** $\Phi(\vec{a}'_k(t)) = \delta(f(\vec{x}'_k(t)), \kappa_k);$
>   *select:* $P(t+1) := s_{\{q\}}(P(t) \cup P'(t));$
> $t := t + 1;$
> **od**

2.2.6 Theory

In his thesis, D. B. Fogel analyzes a standard EP algorithm with $\Phi(\vec{x}) = \beta \cdot f(\vec{x})$, assuming $\beta = 1$ in general ((Fogel 1992b), pp. 142–151). The analysis aims at giving a proof of the global convergence with probability one for the resulting algorithm, and the result is derived by defining a Markov chain over the discrete state space that is obtained from a reduction of the abstract search space $\mathbb{R}^n$ to C^n, where $C \subset \mathbb{R}$ denotes the finite set of numbers representable on a digital computer. By combining all possible populations that contain the grid point $\vec{c} \in C^n$ having objective function value $f(\vec{x})$ closest to the true global optimum f^*, he defines an absorbing state the process will ultimately get trapped in (due to the elitist character of selection).

However, this argumentation is, strictly interpreted, a convergence proof for a grid search method. The convergence theorem for the (1+1)–ES can easily be transferred to standard EP and allows to fill the gap between C^n and $\mathbb{R}^n$.

In case of the simplified sphere model

$$\tilde{f}_2(\vec{x}) \;=\; \sum_{i=1}^{n} x_i^2 = r^2 \;, \tag{15}$$

the convergence rate theory can also be transferred from (1+1)–ES to (1+1)-EP (for population size $\mu = 1$, selection becomes deterministic in EP; see also (Fogel 1992b), pp. 147–150), resulting in a standard deviation for EP on the sphere model of

$$\sqrt{\beta \tilde{f}_2(\vec{x})} \;=\; r\sqrt{\beta} = \frac{n\sigma_2^* \sqrt{\beta}}{1.224} \;. \tag{16}$$

For $\beta = 1$, it can be shown that the convergence rate $\varphi_2(\sigma_2 = r)$ quickly approaches zero as n is increased, while on the other hand a choice $\beta = n^{-2}$ yields a nearly optimal convergence rate. The latter choice was suggested by D. B. Fogel for $n > 2$ (personal communication, August 1992). This clarifies again the importance of a clever step-size control (either by the 1/5-success rule or even better by self-adaptation) especially for increasing problem dimension.

2.3 Genetic Algorithms

GAs are probably the best known Evolutionary Algorithms, receiving remarkable attention all over the world. J. Holland's research interests in the sixties were devoted to the study of general adaptive processes, concentrating on the idea of a system receiving sensory inputs from the environment by binary detectors (Holland 1962; Holland 1975). Structures in the search space were progressively modified in this model by operators selected by an *adaptive plan*, judging about the quality of previous trials by means of an evaluation measure. In (Holland 1975), he points out how to interpret the so-called *reproductive plans* in terms of genetics, economics, game-playing, pattern recognition, and parameter optimization ((Holland 1975), chapter 3). His genetic plans or Genetic Algorithms were applied to parameter optimization for the first time by K. De Jong (De Jong 1975), who laid the foundations of this application technique. Nowadays, numerous modifications of the original GA, usually referred to as the *canonical* GA, are applied to all (and more) fields Holland had indicated. However, many of these applications show enormous differences to the canonical GA as explained in the following, such that the boundary to the other algorithms discussed above becomes blurred. Important examples of non-canonical GAs include D. Whitley's GENITOR system (Whitley 1989), J. Grefenstette's SAMUEL system (Gordon and Grefenstette 1990), and L. Davis' GAs (Davis 1991).

2.3.1 Fitness Evaluation and Representation

As indicated, canonical GAs work on bit strings of fixed length l, i.e. $I = \{0,1\}^l$. For pseudoboolean objective functions, this representation can be used directly. In order to apply canonical GAs to continuous parameter optimization problems of the form $f : \prod_{i=1}^{n} [u_i, v_i] \to \mathbb{R}$ ($u_i < v_i$), the bit string is logically divided into n segments of (in most cases) equal length l_x (i.e. $l = n \cdot l_x$), and each segment is interpreted as the binary code of the corresponding object variable $x_i \in [u_i, v_i]$. A segment decoding function $\Gamma^i : \{0,1\}^{l_x} \to [u_i, v_i]$ typically looks like

$$\Gamma^i(a_{i1} \dots a_{il_x}) \quad = \quad u_i + \frac{v_i - u_i}{2^{l_x} - 1} \left(\sum_{j=1}^{l_x} a_{ij} 2^{j-1} \right) \quad , \tag{17}$$

where $(a_{i1} \dots a_{il_x})$ denotes the i-th segment of an individual $\vec{a} = (a_{11} \dots a_{nl_x}) \in I^{n \cdot l_x} = I^l$. Nowadays, instead of the simple binary code a *Gray code* interpretation of the bit string is normally used for decoding purposes. Combining the segment-wise decoding functions Γ^i to an individual-decoding function $\Gamma = \Gamma^1 \times \dots \times \Gamma^n$, fitness values are obtained by setting $\Phi(\vec{a}) = \delta(f(\Gamma(\vec{a})))$, where again δ denotes a scaling function assuring positive fitness values such that the best individual receives largest fitness. Most commonly, a *linear scaling* is used which takes into account the worst individual of the population $P(t-\omega)$ ω time steps before $(t - \omega < 0 \Rightarrow P(t-\omega) := P(0))$:

$$\delta(f(\Gamma(\vec{a})), P(t-\omega)) \quad = \quad \max\{f(\Gamma(\vec{a}_j)) \mid \vec{a}_j \in P(t-\omega)\} - f(\Gamma(\vec{a})) \quad , \tag{18}$$

where ω is called the *scaling window*. Some other scaling methods are discussed by Goldberg ((Goldberg 1989), pp. 122–124). As an important remark, we emphasize on the fact that this representation method is a special technique developed for the application of canonical GAs to parameter optimization problems. The wide range of alternative representations based on the binary code allows of the application of canonical GAs to a variety of different problem domains.

In terms of molecular genetics, there is an analogy to the discrete alphabet of nucleotide bases, double strands of which form the DNA. This information chain (the *genotype*) is decoded in several steps to amino acids (by means of the genetic code), to proteins, and finally to the phenotype by means of the epigenetic apparatus. Therefore, the GA representation scheme can be interpreted to be slightly closer to the natural model than both other algorithms discussed here. However, the impact of this additional representational level on the suitability for parameter optimization has still to be investigated further. The genetic code as well as the epigenetic apparatus are completely ignored until now.

2.3.2 Mutation

Mutation in canonical GAs works on the bit string level and is traditionally referred to as a "background operator" (Holland 1975). It works by occasionally inverting single bits of individuals, with the probability p_m of this event usually being very small ($p_m \approx 1 \cdot 10^{-3}$ per bit). Often, p_m neither depends on the number n of object variables, nor on the total length l of the bit string. This kind of mutation is ruled by pure randomness like the Monte Carlo method, one cannot speak of a (mean) step size. On a single individual, mutation $m'_{\{p_m\}} : I \to I$, $m'_{\{p_m\}}(s_1, \dots, s_l) = (s'_1, \dots, s'_l)$ works as follows ($\forall i \in \{1, \dots, l\}$):

$$s'_i \quad = \quad \begin{cases} s_i & , \ \chi_i > p_m \\ 1 - s_i & , \ \chi_i \leq p_m \end{cases} \quad . \tag{19}$$

Here $\chi_i \in [0,1]$ is a uniform random variable, sampled anew for each bit. The reader should be very conscious about mutation rate values given in literature, because originally mutation was defined as a substitution of a bit by a random element from the alphabet $\{0,1\}$ (Holland 1975), such that mutation rates according to our definition are twice as large as mutation rates according to Holland's original one. However, because it is more appropriate to interpret mutation as a real change (instead of a random toss with probability one-half), it is used as an inversion event by most GA researchers.

2.3.3 Recombination

In canonical GAs, emphasis is mainly concentrated on *crossover*, the recombination operator of GAs, as the main variation operator which hopefully recombines useful segments from different individuals. Crossover $r'_{\{p_c\}} : I^\mu \rightarrow I$ is again an operator working entirely on the bit representation, completely ignoring the genetic code and the epigenetic apparatus. It does not respect the semantic boundaries l_x of the encoded variables. An exogenous parameter p_c (crossover rate) indicates the probability per individual to undergo recombination. Typical values for p_c are in the range $[0.6, 1.0]$. When two parent individuals $\vec{s} = (s_1, \ldots, s_l)$, $\vec{v} = (v_1, \ldots, v_l)$ have been selected (at random) from the population, crossover forms two offspring individuals $\vec{s}\,'$, $\vec{v}'$ according to the following scheme:

$$\begin{aligned} \vec{s}\,' &= (s_1, \ldots, s_{\chi-1}, s_\chi, v_{\chi+1}, \ldots, v_l) \\ \vec{v}' &= (v_1, \ldots, v_{\chi-1}, v_\chi, s_{\chi+1}, \ldots, s_l) \quad . \end{aligned} \tag{20}$$

As before, $\chi \in \{1, \ldots, l\}$ denotes a uniform random variable, and one of both offspring individuals is randomly selected to be the overall result of crossover. This *one-point crossover* can be extended naturally to a generalized m-point crossover by sampling more than one breakpoint and alternately exchanging each second resulting segment (De Jong 1975). The *uniform crossover* operator drives the number of crossover points to an extreme by performing the random decision whether to exchange information between parents or not for each bit of the genotype anew (Syswerda 1989). On the genotype level this operator corresponds well to discrete recombination in ESs. Actually, however, neither a clear theoretical nor empirical evidence exists to decide upon the question which crossover operator is most appropriate, although several investigations tried to shed light on these questions (Caruna et al. 1989; Eshelman et al. 1989; Schaffer et al. 1989).

2.3.4 Selection

Just as in EP, selection in canonical GAs emphasizes a probabilistic survival rule mixed with a fitness dependent chance to have (different) partners for producing more or less offspring. By deriving an analogy to the game-theoretic *multi-armed bandit problem*, Holland identifies a necessity to use *proportional selection* in order to optimize the tradeoff between further exploiting promising regions of the search space while at the same time also exploring other regions (see (Holland 1975), chapter 5). For proportional selection $s : I^\mu \rightarrow I^\mu$, the reproduction probabilities of individuals $\vec{a}_i$ are given by their relative fitness, i.e. $(\forall i \in \{1, \ldots, \mu\})$:

$$p_s(\vec{a}_i) \quad = \quad \frac{\Phi(\vec{a}_i)}{\sum_{j=1}^{\mu} \Phi(\vec{a}_j)} \quad . \tag{21}$$

Sampling μ individuals according to this probability distribution yields the next generation of parents. Obviously, this mechanism fails in the case of negative fitness or minimization tasks, which explains the necessity to introduce a scaling function δ as mentioned in 2.3.1.

2.3.5 Conceptual Algorithm

As before, the canonical GA is embedded here in the framework of our conceptual algorithm:

ALGORITHM 4 (Outline of a canonical GA)

$t := 0$;
initialize $P(0) := \{\vec{a}_1(0), \ldots, \vec{a}_\mu(0)\} \in I^\mu$
  where $I = \{0,1\}^l$;
evaluate $P(0): \ \{\Phi(\vec{a}_1(0)), \ldots, \Phi(\vec{a}_\mu(0))\}$
  where $\Phi(\vec{a}_k(0)) = \delta(f(\Gamma(\vec{a}_k(0))), P(0))$;
while $(\iota(P(t)) \neq$ **true**$)$ **do** % while termination criterion not fulfilled
  recombine: $\vec{a}'_k(t) := r'_{\{p_c\}}(P(t)) \ \forall k \in \{1, \ldots, \mu\}$;
  mutate: $\vec{a}''_k(t) := m'_{\{p_m\}}(\vec{a}'_k(t)) \ \forall k \in \{1, \ldots, \mu\}$;
  evaluate: $P''(t) := \{\vec{a}''_1(t), \ldots, \vec{a}''_\mu(t)\}$:
    $\{\Phi(\vec{a}''_1(t)), \ldots, \Phi(\vec{a}''_\mu(t))\}$ **where** $\Phi(\vec{a}''_k(t)) = \delta(f(\Gamma(\vec{a}''_k(t))), P(t-\omega))$;
  select $P(t+1) := s(P''(t))$
    where $p_s(\vec{a}''_k(t)) = \Phi(\vec{a}''_k(t)) / \sum_{j=1}^{\mu} \Phi(\vec{a}''_j(t))$;
  $t := t + 1$;
od

Though the order of steps in the main loop does not exactly correspond to the order as defined by Holland, i.e. *select*, *recombine*, *mutate*, *evaluate* (Holland 1975), the difference caused by the formulation given here is likely to be marginal (practically it is just one missing *select* operation after initialization and first evaluation). Experimental runs obtained with both alternative variants did not show statistically relevant differences.

2.3.6 Theory

The essentials of GA-theory were derived by viewing at a canonical GA as an algorithm that processes *schemata*. A schema $H \in \{0, 1, *\}^l$ is a description of a similarity template or hyperplane in l-dimensional bit space. *Instances* of a schema H are all bit strings $a \in \{0,1\}^l$ which are identical to H in all positions where H has a value of 0 or 1 (e.g. $H = (01*1*)$ has four instances: (01010), (01011), (01110), (01111)). Let $m(H^t)$ denote the number of instances of a schema H^t in the population $P(t)$. Furthermore, $o(H)$ denotes the *order* of H, i.e. the number of fixed positions, and $\delta(H)$ denotes its *defining length*, i.e. the distance between its first and last fixed position (e.g. $o(0{*}{*}1{*}) = 2$, $\delta(0{*}{*}1{*}) = 3$). Using the *average schema fitness*

$$f(H^t) = \frac{1}{m(H^t)} \sum_{a_i \in H^t \cap P(t)} f(a_i) \tag{22}$$

of schema H^t in population $P(t)$ and the average fitness $\bar{f}^t$ of $P(t)$, the schema growth equation for proportional selection turns out to be $m(H^{t+1}) = m(H^t) \cdot f(H^t)/\bar{f}^t$. Under the assumption that H^t is above average, i.e. $f(H^t) = \bar{f}^t + c\bar{f}^t$ with a constant value of $c > 0$, after t time steps starting with $t_0 = 0$, we obtain:

$$m(H^t) = m(H^0) \cdot (1 + c)^t \quad . \tag{23}$$

This means, that proportional selection allocates exponentially increasing (decreasing) numbers of trials to above (below) average schemata.

So far, recombination and mutation are not incorporated into the analysis. This is done by calculating the survival probabilities of a schema H^t under simple crossover (which is $1 - p_c \cdot \delta(H^t)/(l-1)$) and mutation ($(1-p_m)^{o(H^t)}$), resulting in the *Schema Theorem* of canonical GAs:

$$m(H^{t+1}) \;\; \geq \;\; m(H^t) \cdot \frac{f(H^t)}{\bar{f}^t} \cdot \left(1 - p_c \frac{\delta(H^t)}{l-1}\right) (1 - p_m)^{o(H^t)} \tag{24}$$

The schema theorem states, that short, low-order, above-average schemata (*building blocks*) receive exponentially increasing trials in the following generations. The choice of a binary alphabet for encoding maximizes the number of schemata processed by a canonical GA and therefore supports the hyperplane sampling process. Building blocks and their combination to form longer and longer useful substrings are the most important working mechanism of a canonical GA. Therefore, consideration of the schema theorem and the building block hypothesis are the main design criteria for applying a canonical GA to a certain problem (Goldberg 1989).

Besides the fact that there is no observer in GAs looking for hyperplane fitnesses, finite populations often do not contain all instances of a specific schema. Observed schema fitnesses thus might quite mislead the search process. Hyperplanes may well contain very good and very bad solutions of the problem at the same time, but this depends on the problem at hand together with the encoding of the decision variables. Objective functions that mislead a canonical GA, so-called *deceptive problems*, are therefore an important field of research in GA-theory (Goldberg 1987; Goldberg et al. 1992; Page and Richardson 1992).

3 Experimental Results

Using a small test set of three objective functions, an experimental comparison of the algorithms was performed. The test functions are representatives of the classes of unimodal, multimodal, and discontinuous functions, the latter one being equiped with plateaus that do not guide the search by local gradient information. For unimodal functions emphasis is laid on convergence velocity while for multimodal functions convergence reliability is critical. These contradictory requirements reflect the trade-off between exploitative (path-oriented) and explorative (volume-oriented) search. The simplified sphere model $\tilde{f}_2$ (equation (15)), its discretization

$$f_3(\vec{x}) = \sum_{i=1}^{n} \lfloor x_i + 0.5 \rfloor^2 , \tag{25}$$

and f_9, a generalized variant of a multimodal function by Ackley ((Ackley 1987), pp. 13–14), i.e.

$$f_9(\vec{x}) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e , \tag{26}$$

are used with $n = 30$ and a feasible region defined by $-30 \le x_i \le 30$ $(\forall i \in \{1, \ldots, n\})$. The function f_9 has been transformed (by addition of the term $20+e$) such that the global minimum point is located at the origin with a function value of zero. The test functions have been selected from a larger test suite without index changes, resulting in the name convention ($\tilde{f}_2$, f_3, f_9) as used here.

The following (more or less) standard parameterizations of the algorithms were used for experimental test runs:

- Evolution Strategy: (30,200)-ES with self-adaptation of $n_\sigma = 30$ standard deviations, no correlated mutations, discrete recombination on object variables and global intermediate recombination on standard deviations. The standard deviations are initialized to a value of 3.0. This variant is denoted ES_{30} in the following.

- Evolution Strategy: (30,200)-ES with self-adaptation of $n_\sigma = 1$ standard deviation, no correlated mutations, no recombination and standard deviations initialized to 3.0. This variant is denoted ES_1.

- Evolutionary Programming: Meta-EP with self-adaptation of $n_\nu = 30$ variances, population size $\mu = 200$, tournament size $q = 10$ for selection, $\alpha = 6$ and an upper bound $c = 25$ for initialization of variances (see (Fogel 1992b), p. 168, p. 173)[1].

- Genetic Algorithm: Population size $\mu = 200$, mutation rate $p_m = 0.001$, crossover rate $p_c = 0.6$, two-point crossover (De Jong 1975), Gray code, and bit string length $l = 32n$ $(= 960)$[2].

Population size settings are oriented towards a comparability of the results with respect to the number of objective function evaluations, which is controlled by the size $\lambda = 200$

[1] I.e. variances are initialized at random in the range $[0, 25]$. EP and ESs are handled differently with respect to this topic in order to be as close to the proposed parameterizations as possible.

[2] 32 bits per object variable are used in order to obtain a maximum resolution $\Delta x_i = (v_i - u_i)/(2^{32} - 1)$ of the search grid.

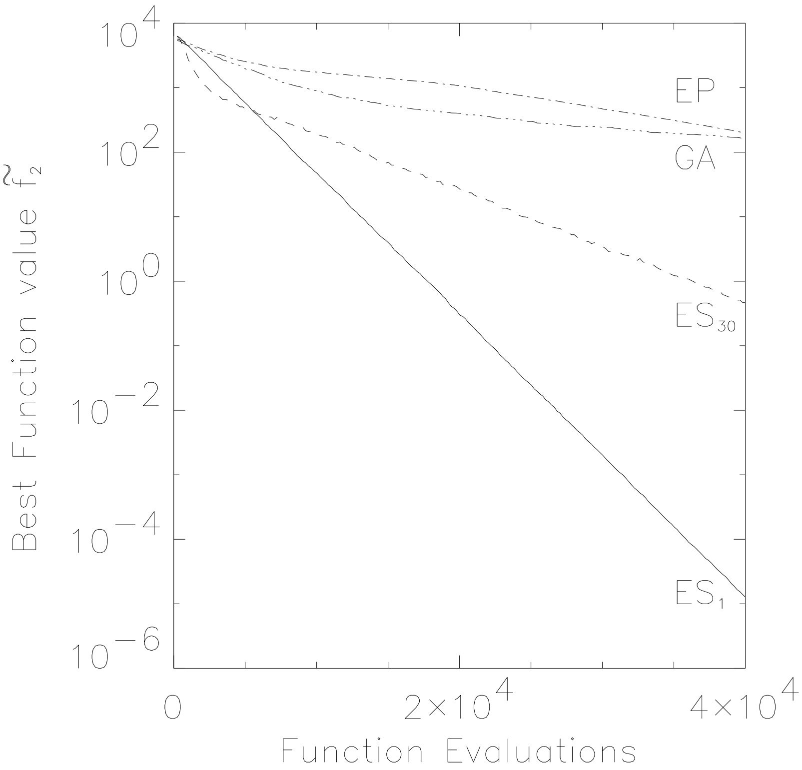

Figure 1: Comparison of performance for runs of Evolutionary Algorithms on $\tilde{f}_2$.

of the offspring population. Since in EP a population size of $\mu = \lambda = 200$ is always used, the other algorithms were adapted to this setting. All results were obtained by running 20 experiments per algorithm and averaging the resulting data. For the sphere model, 40,000 function evaluations were performed for each run, and in case of the discretization f_3 and Ackley's function this was increased to 100,000 function evaluations in order to identify runs that stagnate in contrast to those that approach the global optimum.

Of course, a comparison on just three objective functions does not yield a general assessment of the behavior of Evolutionary Algorithms. For a more general picture, additional important classes of topological characteristics (noise, discontinuities, irregularities of the locations of local optima, narrow valleys, sharp peaks) and a range of different problem dimensions (e.g. $n \in [5, 1000]$) should be considered. Such a deeper experimental investigation is an important topic of further research.

Furthermore, there are some doubts whether the comparison can be fair, since it can be argued that ESs and EP are Evolutionary Algorithms which are relatively specialized to parameter optimization while a canonical GA with its binary encoding mechanism can be conceived of as a more general purpose algorithm.

However one may think about these questions, the results discussed in the following

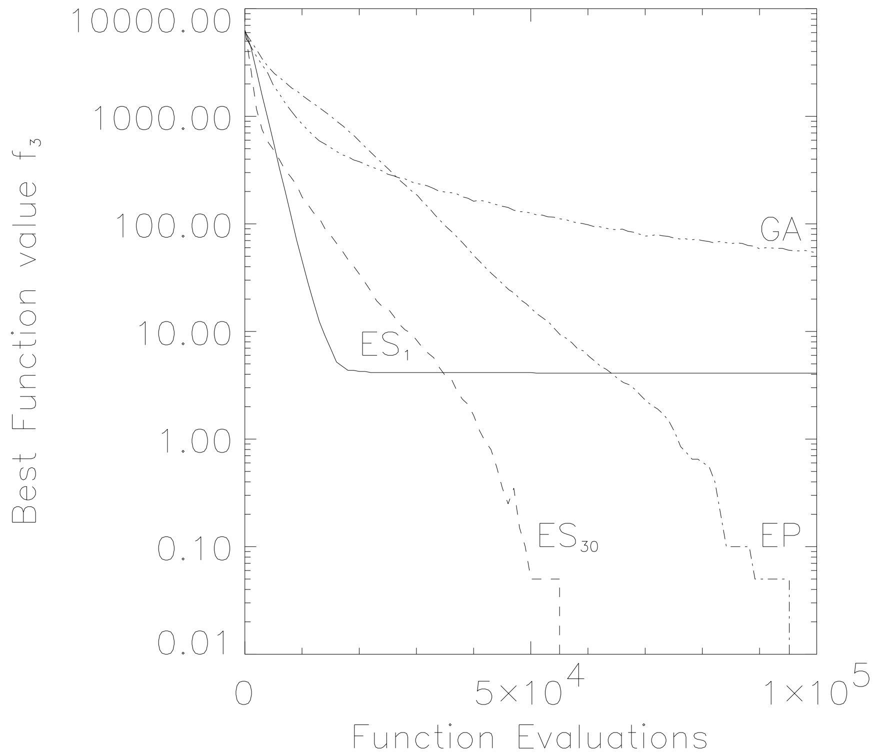

Figure 2: Comparison of performance for runs of Evolutionary Algorithms on f_3.

are intended to provide just an impression of the behavioral differences of ESs, EP, and canonical GAs on a small set of test cases.

First, we discuss the results obtained for the sphere model as shown in figure 1, where the actually best objective function value is plotted over the number of function evaluations. Both variants of the ES show linear convergence, clearly demonstrating its capability to approach a single optimum quickly. The difference in convergence velocity of both variants reflects the large amount of additional information that must be learned when thirty standard deviations rather than one are used. For the totally symmetric, unimodal sphere model these additional degrees of freedom are unnecessary, such that no benefit can be expected from the variant ES_{30} that is more complex than needed. The same argument holds for EP, where seemingly the self-adaptation process works slower than for an ES that self-adapts the same amount of strategy parameters. The difference is likely to be caused by the absence of recombination in EP. The canonical GA applied to the sphere model gives a demonstration of its missing emphasis on convergence velocity and local optimization.

For the step function f_3 the results shown in figure 2 were obtained. For this function, the self-adaptation capabilities of the ES_1 algorithm are not sufficient to locate the globally optimal solution, and the search stagnates completely after an initial phase of rapid progress. Both the ES_{30} and EP have no difficulties in this case and locate the optimal plateau in each run, reflecting the good chance of leaving suboptimal plateaus due to their additional improvement capabilities introduced by thirty independently variable step sizes. The behavior of the canonical GA is almost identical to that obtained on $\tilde{f}_2$, i.e. the algorithm is “unimpressed” by the introduction of plateaus and discontinuities, but it remains the slowest of the three algorithms compared here.

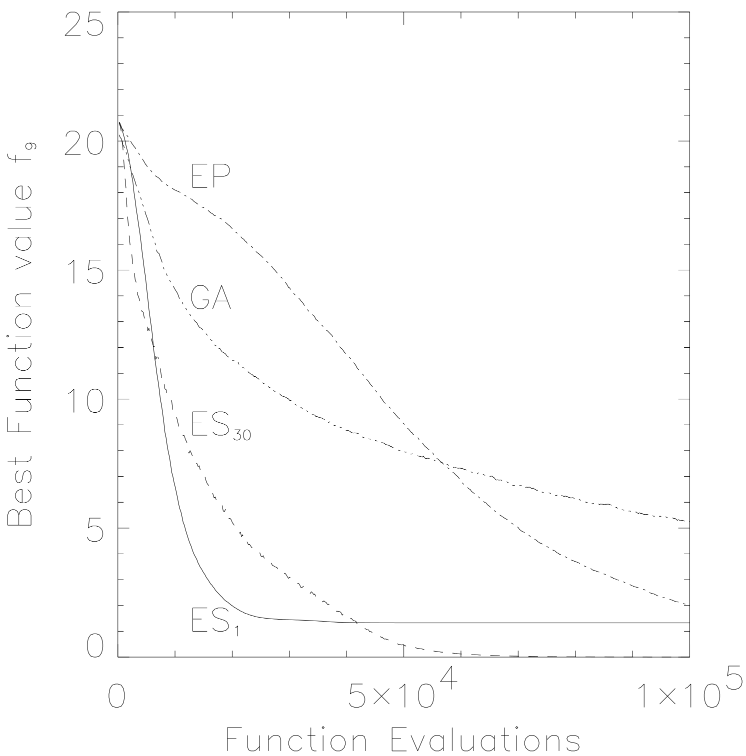

Figure 3: Comparison of performance for runs of Evolutionary Algorithms on f_9.

A completely different behavior is observed on the continuous, multimodal objective function f_9, for which the experimental data is shown in figure 3. Here, convergence reliability is the criterion that determines the quality of an algorithm, and the advantages of self-adapting thirty standard deviations in ESs are again clearly identified in the graphic, since in all twenty experiments performed the ES_{30} variant located the global optimum (achieving objective function values better than 10^{-4}). In contrast to this, in fourteen of twenty experiments an ES using just one mutation rate got trapped in local optima as reflected by the stagnating curve for ES_1 shown in figure 3 (the other six runs located the global optimum). Experimental runs of more than 100,000 function evaluations would be necessary to assess the final results of EP and the canonical GA, but the trend allows of predicting a reasonable convergence reliability also for these algorithms in the long run. Within the twenty runs, EP identified the global optimum in one case.

Table 1 summarizes the mean best objective function values within the last generation of the runs and their standard deviations (from twenty experiments) for the three functions. The small standard deviation of ES_{30} on f_9 reflects again the high convergence reliability of this ES variant, while standard deviations of the other variants, including ES_1, indicate the large diversity of locally optimal solutions. On the sphere model, standard deviations and results for both EP and the canonical GA demonstrate that these strategies are some orders of magnitude slower than the ES. The results for the step function are added for reasons of completeness.

	$\tilde{f}_2$		f_3		f_9	
	Mean best	St. dev.	Mean best	St. dev.	Mean best	St. dev.
ES_1	$1.075 \cdot 10^{-5}$	$4.730 \cdot 10^{-6}$	4.100	3.177	1.326	1.039
ES_{30}	$6.672 \cdot 10^{-1}$	$2.610 \cdot 10^{-1}$	0	0	$1.618 \cdot 10^{-3}$	$9.290 \cdot 10^{-4}$
EP	$1.998 \cdot 10^{2}$	$6.564 \cdot 10^{1}$	0	0	1.976	$6.300 \cdot 10^{-1}$
GA	$1.647 \cdot 10^{2}$	$4.741 \cdot 10^{1}$	$5.390 \cdot 10^{1}$	$1.286 \cdot 10^{1}$	5.253	$5.130 \cdot 10^{-1}$

Table 1: Mean best objective function values after $40,000$ ($\tilde{f}_2$) / $100,000$ (f_3, f_9) function evaluations and corresponding standard deviations for the three objective functions. Results are averaged over 20 experiments.

These experiences do not allow for drawing general conclusions. On the small test set discussed here, the combination of self-adaptation, recombination, and relatively strong selective pressure as used in ESs has some advantages both for convergence velocity and convergence reliability. Though the general trade-off between these contradicting requirements can not be avoided, a comparison to canonical GAs, where self-adaptation and high selective pressure are missing, and EP, where recombination and high selective pressure are missing, gives some hints that the trade-off can best be treated by the collective application of these principles.

For canonical GAs, it has been demonstrated in different papers that they can be turned into more powerful parameter optimization procedures by incorporating self-adaptation and increasing selective pressure (see (Bäck and Hoffmeister 1991; Bäck 1992)). However, the discussion whether Genetic Algorithms should emphasize on their parameter optimization properties or alternatively on their more general adaptive capabilities is still ongoing (see e.g. (De Jong 1992)), and it is clear that they benefit from a broad range of possible applications even in combinatorial optimization (due to the discrete nature of the code).

Similarly, recombination and higher selective pressure may also be useful when incorporated into EP, since the role of recombination on strategy parameters for supporting a successful self-adaptation has not been tested in EP so far (in contrast to recombination on object variables, which was identified in (Fogel and Atmar 1990), using just one objective function, to be unnecessary for successful optimization — a result that may be worth further examination).

4 Summary

The main characteristic similarities and differences of the algorithms presented in this article are summarized in table 2.

	ES	EP	GA
Representation	Real-valued	Real-valued	Binary-valued
Self-adaptation	Standard deviations and covariances	Variances (in meta-EP)	None
Fitness is	Objective function value	Scaled objective function value	Scaled objective function value
Mutation	Main operator	Only operator	Background operator
Recombination	Different variants, important for self-adaptation	None	Main operator
Selection	Deterministic, extinctive	Probabilistic, extinctive	Probabilistic, preservative

Table 2: Main characteristics of Evolutionary Algorithms.

It is a remarkable fact that each algorithm emphasizes different features as being most important for a successful evolution process. In analogy to repair-enzymes which give evidence for a biological self-control of mutation rates of nucleotide bases in DNA, both ESs and meta-EP use self-adaptation processes for the mutation rates. In canonical GAs, this concept was successfully tested only recently (Bäck 1992), but will need more time to be recognized and applied. Both ESs and EP concentrate on mutation as the main search operator, while the role of (pure random) mutation in canonical GAs is usually seen to be of secondary (if any) importance. On the other hand, recombination plays a major role in canonical GAs, is missing completely in EP, and is urgently necessary for use in connection to self-adaptation in ESs. One of the characteristics of EP is the strict denial of recombination being important for the search. Finally, both canonical GAs and EP emphasize on a necessarily probabilistic selection mechanism, while from the ESs point of view selection is completely deterministic without any evidence for the necessity of incorporating probabilistic rules. In contrast, both ESs and EP definitely exclude some individuals from being selected for reproduction, i.e. they use *extinctive* selection mechanisms, while canonical GAs generally assign a nonzero selection probability to each individual, which we term a *preservative* selection mechanism.

Very naturally, the reader can deduce several interesting questions for future research. It is curious enough to see very different, sometimes contrasting design principles for Evolutionary Algorithms being emphasized by the different research communities. A clear goal of future research should be to identify the rational as well as not so rational reasons for this fact and to extract the general rules for designing components of new and may be even better EAs.

We conclude by mentioning again that this paper concentrated on the use of Evolutionary Algorithms for solving real-valued parameter optimization problems. An interesting area of research involves understanding the extent to which these ideas can be extended to other problem domains such as optimization problems with non-linear constraints (in ESs, H.-P. Schwefel has solved this problem by repeating the creation and evaluation of offspring individuals as long as constraints are violated (Schwefel 1977) and by introducing correlated mutations (Schwefel 1981)), discrete optimization problems, and problems in which the response surface is changing during the evolutionary process. Examples pointing into these directions are reported in the field of ESs by modeling the concept of *somatic* mutations for solving discrete problems (Schwefel 1975) and by using changing environments and dominance / recessivity for successfully solving problems of multiple criteria decision making (Kursawe 1991).

5 A Guide to Relevant Literature

Since this article contains references to the basic literature on each type of three Evolutionary Algorithms, it seems appropriate to guide the reader by providing a hint to further reading. For each algorithm discussed above, we can identify two generations of books, i.e.:

- ESs: Rechenberg discusses the (1+1)–ES and its theory in (Rechenberg 1973) (only in German language). A look at Schwefel's book (Schwefel 1981), where the $(\mu+\lambda)$–ES and (μ,λ)–ES are introduced and compared to traditional optimization methods, may be more useful. Furthermore, a convergence rate theory for these algorithms is developed for the test functions introduced by Rechenberg.

- EP: For historical reasons it is worth looking at the early work of L. J. Fogel, e.g. (Fogel et al. 1966), since his ideas were vehemently criticized at that time. For the modern EP algorithm, the reader is referred to D. B. Fogel's book on using EP for system identification (Fogel 1991), for meta-EP to his thesis (Fogel 1992b).

- GAs: Holland gives a detailed discussion of adaptive systems in general and theory as well as basic algorithms of a GA in his first book (Holland 1975). The modern state-of-the-art and a detailed historical overview are presented in the more practically oriented book by Goldberg (Goldberg 1989).

For those readers interested in a more detailed, implementation-oriented introduction to the algorithms, we recommend a look at (Schwefel 1981) (ESs), (Fogel 1992b) (EP), and (Goldberg 1989) (GAs). Practical applications and unusual extensions of Genetic

Algorithms are presented in (Davis 1991) and (Michalewicz 1992). Specialized articles can also be found in the conference proceedings (mentioned in section 1). The following — necessarily incomplete — reference list is sorted according to the special type of Evolutionary Algorithm that is mainly discussed and in each category by publication date in order to provide a quick overview of the historical development and literature categories.

Evolution Strategies

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. Roy. Aircr. Establ., libr. transl. 1122. Farnborough, Hants., UK.

Schwefel, H.-P. (1965). *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik.* Diploma thesis, Technical University of Berlin.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart: Frommann–Holzboog.

Schwefel, H.-P. (1975). Binäre Optimierung durch somatische Mutation. Technical report, Technical University of Berlin and Medical University of Hannover.

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, Volume 26 of *Interdisciplinary systems research.* Basel: Birkhäuser.

Schwefel, H.-P. (1981). *Numerical optimization of computer models.* Chichester: Wiley.

Schwefel, H.-P. (1987). Collective phenomena in evolutionary systems. In *Preprints of the 31st annual meeting of the int'l society for general system research, Budapest*, Volume 2, (pp. 1025–1033).

Kursawe, F. (1991). A variant of evolution strategies for vector optimization. In Schwefel and Männer (1991), (pp. 193–197).

Evolutionary Programming

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution.* New York: Wiley.

Fogel, D. B., & Atmar, J. W. (1990). Comparing genetic operators with gaussian mutations in simulated evolutionary processes using linear systems. *Biological Cybernetics 63*, 111–114.

Fogel, D. B. (1991). *System identification through simulated evolution: A Machine Learning Approach to Modeling.* Needham Heights: Ginn Press.

Fogel, D. B. (1992a). An analysis of evolutionary programming. In Fogel and Atmar (1992), (pp. 43–51).

Fogel, D. B., & Atmar, J. W. (Eds.). (1992). *Proceedings of the first annual conference on evolutionary programming.* La Jolla, CA: Evolutionary Programming Society.

Fogel, D. B. (1992b). *Evolving artificial intelligence.* Ph. D. thesis, University of California, San Diego, CA.

Genetic Algorithms

Holland, J. H. (1962). Outline for a logical theory of adaptive systems. *Journal of the Association for Computing Machinery*, *3*, 297–314.

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

De Jong, K. (1975). *An analysis of the behaviour of a class of genetic adaptive systems.* Ph. D. thesis, University of Michigan, Ann Arbor, MI.

Grefenstette, J. J. (Ed.). (1985). *Proceedings of the first int'l conference on genetic algorithms and their applications*, Hillsdale, NJ: Lawrence Erlbaum.

Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics SMC–16*(1), 122–128.

Davis, L. (Ed.) (1987). *Genetic algorithms and simulated annealing.* Research Notes in Artificial Intelligence. London: Pitman.

Goldberg, D. E. (1987). Simple genetic algorithms and the minimal, deceptive problem. In L. Davis (Ed.), *Genetic algorithms and simulated annealing,* (pp. 74–88). London: Pitman.

Grefenstette, J. J. (Ed.) (1987). *Proceedings of the second int'l conference on genetic algorithms and their applications*, Hillsdale, NJ: Lawrence Erlbaum.

Grefenstette, J. J. (1987b). *A user's guide to* GENESIS. Washington, D. C.: Navy Center for Applied Research in Artificial Intelligence.

Caruna, R. A., Eshelman, L. J., & Schaffer, J. D. (1989). Representation and hidden bias II: Eliminating defining length bias in genetic search via shuffle crossover. In N. S. Sridharan (Ed.), *Eleventh int'l joint conference on artificial intelligence*, (pp. 750–755). Morgan Kaufmann.

Eshelman, L. J., Caruana, R. A., & Schaffer, J. D. (1989). Biases in the crossover landscape. In Schaffer (1989), (pp. 10–19).

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Reading, MA: Addison Wesley.

Schaffer, J. D. (Ed.) (1989). *Proceedings of the third int'l conference on genetic algorithms and their applications*, San Mateo, CA: Morgan Kaufmann.

Schaffer, J. D., Caruana, R. A., Eshelman, L. J., & Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. In Schaffer (1989), (pp. 51–60).

Syswerda, G. (1989). Uniform crossover in genetic algorithms. In Schaffer (1989), (pp. 2–9).

Whitley, D. (1989). The GENITOR algorithm and selection pressure: Why rank–based allocation of reproductive trials is best. In Schaffer (1989), (pp. 116–121).

Gordon, D. A., & Grefenstette, J. J. (1990). Explanations of empirically derived reactive plans. In *Proceedings of the seventh int'l conference on machine learning*, (pp. 198–203). Morgan Kaufmann.

Belew, R. K., & Booker, L. B. (Eds.). (1991). *Proceedings of the fourth int'l conference on genetic algorithms and their applications*, San Diego, CA: Morgan Kaufmann.

Bäck, T., & Hoffmeister, F. (1991). Extended selection mechanisms in genetic algorithms. In Belew and Booker (1991), (pp. 92–99).

Davidor, Y. (1991). *Genetic algorithms and robotics. A heuristic strategy for optimization*, Volume 1 of *World scientific series in robotics and automated systems*. Singapore: World Scientific.

Davis, L. (Ed.) (1991). *Handbook of genetic algorithms*. New York: Van Nostrand Reinhold.

Bäck, T. (1992). Self-adaptation in genetic algorithms. In *Proceedings of the first european conference on artificial life*, Paris, France, (pp. 263–271). Cambridge, MA: The MIT Press.

Goldberg, D. E., Deb, K., & Horn, J. (1992). Massive multimodality, deception, and genetic algorithms. In Männer and Manderick (1992), (pp. 37–46).

De Jong, K. (1992). Are genetic algorithms function optimizers ? In Männer and Manderick (1992), (pp. 3–13).

Michalewicz, Z. (1992). *Genetic algorithms + data structures = evolution programs*. Artificial Intelligence. Berlin: Springer.

Page, S. E., & Richardson, D. W. (1992). Walsh functions, schema variance, and deception. *Complex Systems*, *6*, 125–135.

Evolutionary Algorithms and Other Literature

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

Schwefel, H.-P., & Männer, R. (Eds.) (1991). *Parallel problem solving from nature — Proc. 1st workshop PPSN I*, Volume 496 of *Lecture Notes in Computer Science*. Berlin: Springer.

Bäck, T., Hoffmeister, F., & Schwefel, H.-P. (1992). Applications of evolutionary algorithms. (Report of the Systems Analysis Research Group SYS–2/92). University of Dortmund, Department of Computer Science.

Männer, R., & Manderick, B. (Eds.) (1992). *Parallel problem solving from nature, 2*. Amsterdam: Elsevier.

Rudolph, G. (1992). Parallel approaches to stochastic global optimization. In W. Joosen and E. Milgrom (Eds.), *Parallel computing: from theory to sound practice, Proceedings of the european workshop on parallel computing*, Barcelona, Spain, (pp. 256–267). Amsterdam: IOS Press.

Summary of Basic Notation

Notation	Description
I	Space of individuals
$\vec{a} \in I$	A single individual
$\vec{x} \in \mathbb{R}^n$	Vector of object variables
$f : \mathbb{R}^n \to \mathbb{R}$	Objective function
$\Phi : I \to \mathbb{R}$	Fitness function
μ	Parent population size
λ	Offspring population size
$P(t) = \{\vec{a}_1(t), \ldots, \vec{a}_\mu(t)\}$	Population at generation t
$r_{\Theta_r} : I^\mu \to I^\lambda$	Recombination operator (global form)
$r'_{\Theta_r} : I^\mu \to I$	Recombination operator (local form)
$m_{\Theta_m} : I^\lambda \to I^\lambda$	Mutation operator (global form)
$m'_{\Theta_m} : I \to I$	Mutation operator (local form)
$s_{\Theta_s} : (I^\lambda \cup I^{\mu+\lambda}) \to I^\mu$	Selection operator
$\iota : I^\mu \to \{\text{true}, \text{false}\}$	Termination criterion

Table 3: Notational conventions common to all Evolutionary Algorithms described in the paper.

Acknowledgements

The first author gratefully acknowledges financial support by the DFG (Deutsche Forschungsgemeinschaft), grant Schw 361/5-1. Special thanks to David B. Fogel and Kenneth De Jong for helpful discussions about EP and GAs, respectively.