Evolution Strategies: An Alternative Evolutionary Algorithm

Thomas Bäck

Informatik Centrum Dortmund
Joseph-von-Fraunhofer-Str. 20
D-44227 Dortmund

Abstract. In this paper, *evolution strategies* (ESs) — a class of evolutionary algorithms using normally distributed mutations, recombination, deterministic selection of the $\mu > 1$ best offspring individuals, and the principle of self-adaptation for the collective on-line learning of strategy parameters — are described by demonstrating their differences to *genetic algorithms*. By comparison of the algorithms, it is argued that the application of canonical genetic algorithms for continuous parameter optimization problems implies some difficulties caused by the encoding of continuous object variables by binary strings and the constant mutation rate used in genetic algorithms. Because they utilize a problem-adequate representation and a suitable self-adaptive step size control guaranteeing linear convergence for strictly convex problems, evolution strategies are argued to be more adequate for continuous problems.
The main advantage of evolution strategies, the self-adaptation of strategy parameters, is explained in detail, and further components such as recombination and selection are described on a rather general level.
Concerning theory, recent results regarding convergence velocity and global convergence of evolution strategies are briefly summarized, especially including the results for (μ,λ)-ESs with recombination. It turns out that the theoretical ground of ESs provides many more results about their behavior as optimization algorithms than available for genetic algorithms, and that ESs have all properties required for global optimization methods. The paper concludes by emphasizing the necessity for an appropriate step size control and the recommendation to avoid encoding mappings by using a problem-adequate representation of solutions within evolutionary algorithms.

1 Optimization and Genetic Algorithms

In contrast to the title and the overall intention of this article to provide an overview of *evolution strategies* (ESs) [35, 36, 40, 44] (and, to make the reader curious of reading more about them), I take the freedom to start with a brief look at the global optimization problem and those evolutionary algorithms which are widely used to approximately solve this problem: *Genetic algorithms* (GAs) [21, 25]. Although these two different, independently developed branches of evolutionary computation are known since more than thirty years, genetic algorithms have gained much more interest during the past ten years than evolution strategies

did. *Evolutionary programming* (EP) [18, 20], the third main stream evolutionary algorithm, has strong similarities to evolution strategies and could be discussed here as well — but this is more adequately done by David B. Fogel, whose contribution in this volume is a must for every reader interested in evolutionary computation.

In the following, I will assume some basic familiarity with the principles of organic evolution, namely, a *population* of *individuals* which are subject to processes of *mutation*, *recombination*, and *selection*. On a higher level of abstraction, the notion of different *species* and the interactions between species are also incorporated into the algorithms. Furthermore, I also take the freedom to assume that genetic algorithms in their canonical form — using a binary representation of individuals as fixed length strings over the alphabet $\{0, 1\}$, a mutation operator that occasionally inverts single bits with a small probability p_m, a crossover operator that exchanges substrings between individuals with crossover probability p_c, and a proportional selection operator that selects individuals probabilistically on the basis of their relative fitness $f_i / \sum_{j=1}^{\lambda} f_j$ — are known to the reader. If this is not the case, the following should be sufficiently general to provide at least an impression of the major principles and differences between the algorithms, or the reader might wish to consult a more technical overview article such as [10] or the in-depth introductions to evolutionary computation [4, 18].

Though it is often claimed that genetic algorithms are developed for solving *adaptation* rather than *optimization* problems (see e.g. [26, 27]), the distinction (if any) between both terms is neither obvious nor formally defined. Biologically, adaptation denotes a general advantage in efficiency of an individual over other members of the population, and the process of attaining this state (see [31], pp. 134–135). This implies an interpretation of adaptation as optimization in a changing environment — a problem which is handled well by GAs, ESs, and EP as well. In fact, most practical applications of GAs exploit their optimization capabilities (see the bibliography of Alander [1] for a collection of numerous applications of evolutionary algorithms, especially GAs). It is even more surprising to see that many practical applications of GAs do not aim at solving strictly pseudoboolean optimization problems

$$f : M \subseteq \{0, 1\}^{\ell} \to I\!R \quad , \tag{1}$$

(which arise for a variety of combinatorial problems such as knapsacks [30], scheduling [29], graph problems [7, 28], and others), but rather at solving problems of the form

$$f : M \subseteq I\!R^n \to I\!R \quad , \tag{2}$$

which are naturally defined over a continuous, real-valued search space. To apply GAs in their canonical form, a suitable encoding of real-valued vectors $\mathbf{x} \in I\!R^n$ as binary strings $\mathbf{y} \in \{0, 1\}^{\ell}$ is required, and typically this is achieved by subdividing $\mathbf{y}$ into n segments of equal length, decoding each segment to yield the corresponding integer value (either using the standard binary code or a Gray code), and mapping the integer value linearly to an interval $[u_i, v_i] \subset I\!R$ of real values. Figure 1 illustrates this decoding process for string segments of length

nine bits (allowing for the representation of the integers $\{0, 1, \ldots, 511\}$), which are mapped to the interval $[-50, 50]$.

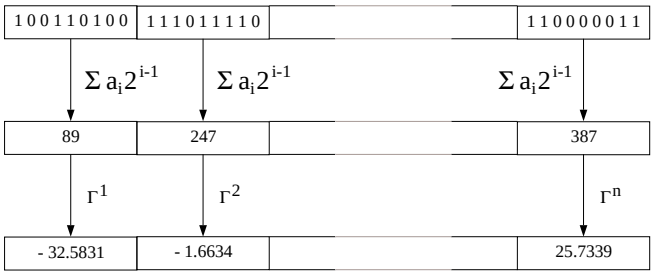

Fig. 1.: Decoding process used in canonical genetic algorithms for continuous search spaces. Γ^i denotes the linear mapping of an integer value $k \in \{0, \ldots, 2^\ell - 1\}$ to the interval $[u_i, v_i]$.

Notice that this representation requires the specification of so-called *box constraints* $[u_i, v_i]$ for the search space. Furthermore, it introduces a discretization of the original search space, and the resolution of the resulting grid of points depends critically on the number of bits used to encode single object variables (I am always critical about applications where less than ten bits are used per object variable — but sometimes, when the precision of the object variables is a priori known to be limited, as in the case of some high energy physics experiments, this might also be an invaluable advantage [23]). The discretization of the search space introduced by the binary code implies that the genetic algorithm might fail to locate an optimal solution exactly just because this solution is not represented by any of the binary strings.

An other disadvantage (or advantage ?) of the binary code is that by mutation of a single bit, depending on its significance in the code, an arbitrarily large modification of the corresponding integer value might result. No preference of small phenotypic changes over large ones — as common in nature — is realized in canonical GAs for parameter optimization tasks. Only few properties of the binary versus Gray code and its impact on the search in combination with the mutation and recombination operators are known and provide some hints towards an explanation of the fact that canonical GAs are sometimes quite useful and the Gray code is typically observed to yield better results than the standard binary code [14, 24]:

- The Gray code and bitwise mutation introduces a probability density function in the decoded object variable space that prefers smaller modificiations over large ones and therefore has some basic similarity to the normal distribution in ESs [4, 24].

– The Gray code does not introduce additional multimodality on the level of the decoding function, as the standard code typically does [3].

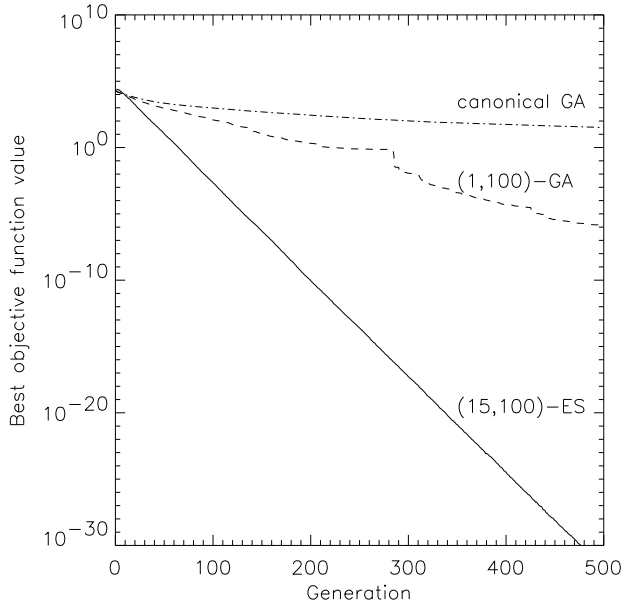

Fig. 2.: Comparison of the convergence velocity of a (15,100)-ES with $n_\sigma = 1$, a (1,100)-GA with a mutation rate $p_m = 0.001$, and a canonical GA with $p_m = 0.001$, $p_c = 0.6$, two-point crossover, and proportional selection.

To look for the pure effect of mutation on binary strings in a simple genetic algorithm, I present some experimental results on an objective function which is so simple that nobody would ever apply an evolutionary algorithm to optimize it — but it serves extremely well for experimental investigations regarding the *convergence velocity* of such algorithms: The sphere model $f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$. The experiment compares the convergence velocity, measured in terms of the best objective function value f_t' occurring in the population as a function of the generation counter t, for a canonical GA with population size $\lambda = 100$, a simple (15,100)-ES, and a (1,100)-GA *without recombination*. The notation (15,100) describes the selection scheme, indicating that 15 parent individuals generate 100 offspring individuals, and the 15 best of the offspring individuals are deterministically selected as parents of the next generation. The evolution strategy works with a self-adaptive control of the mutation step size (see the next section for an explanation of this), while the genetic algorithm uses a constant

mutation rate $p_m = 1/\ell$ and a string length $\ell = 30 \cdot n$ in order to obtain a search grid resolution comparable to that of the evolution strategy. A search space dimension $n = 30$ is used in the experiment. The results, averaged over 20 runs per experiment, are shown in figure 2.

Notice that the canonical GA tends to stagnate after a number of generations, while the modified GA, using mutation only, clearly shows a *linear order of convergence*, i.e., $\ln f'_t/f'_0 \propto t$. In fact, although this linear convergence is not comparable to the convergence velocity of an evolution strategy, it may serve as a counterargument against the claim that recombination is the most important search operator in genetic algorithms and mutation plays only a minor role [21, 25] (minimum requirements for evolution are mutation and selection, not recombination !). Using proportional selection rather than (1,100)-selection, however, the canonical genetic algorithm is not able to exploit the innovative, but undirected power of mutation, and its convergence velocity decreases as the optimum is approached, because a favorable mutation becomes more and more unlikely and, at the same time, the preservation of favorable genotypes is quite weak. In fact, in order to approach the optimum further and further, *the GA has to adjust its "step size" of mutation implicitly* by concentrating bit inversion on bits of smaller and smaller significance, while the step size adaptation in an ES works by evolutionary processes of trial and error.

The order of convergence is an important measure for any kind of optimization method, because a sufficiently large order of convergence assures that the algorithm will at least yield a local optimum within a reasonable time — and time is a critical factor for any practical application. While it was recently claimed by Voigt, Mühlenbein and Cvetković that linear convergence is the best one can achieve for evolutionary algorithms [48], Bäck, Rudolph and Schwefel have shown that evolution strategies and evolutionary programming are in fact able to achieve this convergence order [8].

In addition to a linear order of convergence, which assures a sufficiently large velocity, the property of *global convergence* is the second, at least theoretically important requirement for any global optimization algorithm: Given unlimited time, the algorithm has to find a globally optimal solution with probability one, i.e.,

$$\mathcal{P}(\lim_{t \to \infty} \mathbf{x}'_t = \mathbf{x}^*) = 1 \tag{3}$$

($\mathbf{x}'_t$ denotes the best solution occurring at generation t, and $\mathbf{x}^*$ denotes a globally optimal solution, i.e., $f(\mathbf{x}^*) \leq f(\mathbf{x})\ \forall \mathbf{x} \in M$). For *elitist* evolutionary algorithms, where the best parental solution is guaranteed to survive when no offspring individual represents an improvement of the currently best parent, global convergence with probability one holds under rather general conditions, provided the mutation step size is larger than zero (see [37] for GAs, [8] for ESs and EP), but the canonical GA lacks this property [37]. For evolution strategies, Rudolph recently demonstrated that the non-elitist $(1,\lambda)$-strategy converges globally with probability one if the objective function is stricly convex at least in a neighborhood around the globally optimal point [38].

For practical applications, however, it is not the global convergence with probability one that one is most interested in, but rather the capability of the algorithm to find a solution which is better than the solution known so far. In some sense, it is the *practical convergence reliability* of an evolutionary algorithm that we are most interested in.

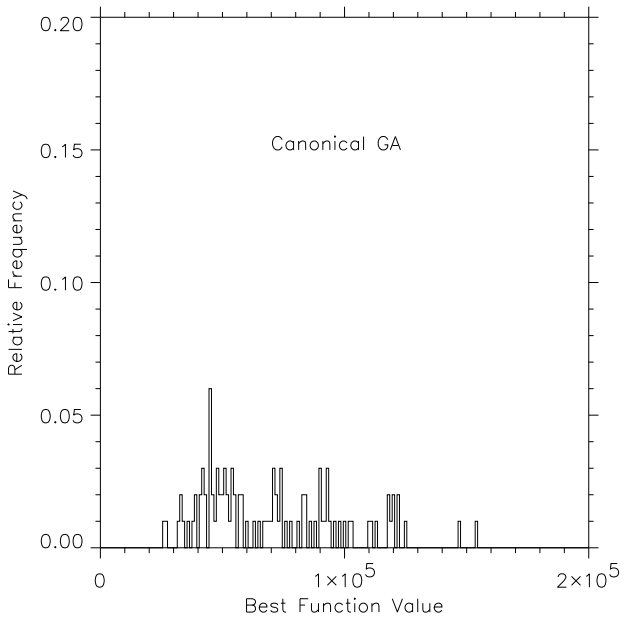

Fig. 3.: Histogram of the final best objective function value obtained from a canonical GA on the function after Fletcher and Powell.

It is almost impossible to assess the general convergence reliability of evolutionary algorithms, and any attempt to do so is necessarily restricted to a limited number of experiments on a limited number of practical problems or test problems. An interesting test problem which was introduced by Fletcher and Powell [16] as a typical representative of nonlinear parameter estimation problems will serve here as an example that reflects my personal experience of the comparative convergence reliability of evolution strategies and genetic algorithms — this assessment is based on personal experience, not on any measurable, general quantity. The Fletcher and Powell function is highly multimodal, and its optima are randomly distributed according to two random $n \times n$ matrices $\mathbf{A} = (a_{ij})$ and

$\mathbf{B} = (b_{ij})$ $(a_{ij}, b_{ij} \in [-100, 100])$, where

$$f_{FP}(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{A}_i - \mathbf{B}_i)^2 \tag{4}$$

$$A_i = \sum_{j=1}^{n} (a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j) \tag{5}$$

$$B_i = \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j)$$

and the global optimum equals the vector $\boldsymbol{\alpha}$, with $f_{FP}(\boldsymbol{\alpha}) = 0$. Up to 2^n extrema are located in the search region $|x_i| \leq \pi$. For the details of the matrices $\mathbf{A}$ and $\mathbf{B}$ as well as the vector $\boldsymbol{\alpha}$, the reader is referred to [4].

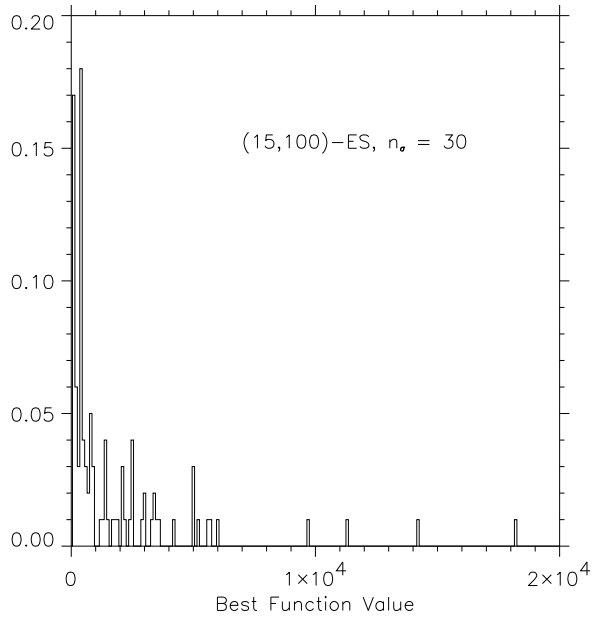

Fig. 4.: Histogram of the final best objective function value obtained from a (15,100)-ES with $n_\sigma = 1$ on the function after Fletcher and Powell.

Since different runs of an evolutionary algorithm on multimodal functions typically stagnate in different local optima, some insight into the reliability to obtain good solutions can be gained from performing a large number of independent runs and looking at the histogram of final solution quality, i.e., the

frequency distribution of the best objective function value found after a certain number of generations. I performed 100 runs over 2000 generations, each, with a canonical GA and the (15,100)-ES (with n self-adaptive step sizes), and the resulting histograms are shown in figure 3 for the genetic algorithm, and in figure 4 for the evolution strategy. Notice that most of the runs of the ES yield an objective function value close to zero, the globally optimal one, and only few outliers with final objective function values above 10^4 can be identified. On the contrary, the canonical GA yields a distribution which is much more spreaded between best values of about $2.5 \cdot 10^4$ and worst values of about $1.5 \cdot 10^5$ (notice that the abscissa axis of figure 3 covers a range of values ten times as large as in figure 4).

As I indicated before, these results are representative of my personal experience, and there is no evidence that evolution strategies would generally outperform genetic algorithms on multimodal problems. To be fair, I have to emphasize that the ES certainly benefits from a larger degree of freedom by working with n different self-adaptive standard deviations per individual in contrast to a single mutation rate in genetic algorithms, exogenously predefined and constant over the complete evolution process.

My conclusions, up to this point, however, are certainly a bit provocative, because they attempt a clear distinction of the preferred application domains of GAs and ESs: *For continuous parameter optimization problems, evolution strategies should be preferred over genetic algorithms, because*

- *the binary search space representation might introduce additional multimodalities and therefore make the search more difficult* [1],
- *the self-adaptation of strategy parameters provides a larger flexibility of the evolution strategy, and*
- *the evolution strategy combines convergence velocity and convergence reliability in a more robust way than genetic algorithms do.*

Presently, some researchers circumvent the coding problem in genetic algorithms by recently developed real-coded "genetic algorithms" (e.g., the real-coded GAs of Davis, which adapt operator probabilities on the basis of the past usefulness of these operators in previous generations [15]), such that the gap between ESs and GAs becomes narrower. Like Michalewicz [32] and Davis [15], I prefer the point of view that the most problem-adequate representation of individuals should be used as a starting point for the development of an evolutionary heuristic.

Besides the representation, step size adaptation over the search process is a topic of major importance. Linear convergence order can be achieved only by an appropriate adjustment of step sizes, and ESs use an elegant way to achieve this: Evolutionary learning, called self-adaptation. This principle, which is not that simple to understand at first glance, is the topic of the next section.

[1] I did not discuss this statement in sufficient detail here, but refer the reader to [3], where the multimodality problem is analyzed.

2 The Tricky Step: Self-Adaptation

In addition to the object variables $x_i \in I\!R$ themselves, also the strategic meta-parameters — e.g., variances and covariances of a normal distribution for mutation — can be learned by evolutionary processes during the search. This is the simple, but strikingly powerful idea of Schwefel[2] [40], to search the space of solutions and strategy parameters in parallel and to enable a suitable adjustment *and* diversity of mutation parameters under arbitrary circumstances. The natural model consists in the existence of repair enzymes and mutator genes coded on the DNA, thus providing partial control of the DNA over its own mutation probability ([22], pp. 269–271).

Technically, this so-called *self-adaptation* principle combines the representation of a solution and its associated strategy parameters within each individual, and the strategy parameters are subject to mutation and recombination just as the object variables. Selection exploits the implicit link between fitness and strategy parameters, thus favoring useful strategy parameters due to their advantageous impact on object variables. In case of evolution strategies, mutation works by adding a normally distributed random vector[3] $\mathbf{z} \sim \mathbf{N}(\mathbf{0}, \mathbf{C})$ with expectation $\mathbf{0}$ and covariance matrix $\mathbf{C}^{-1}$, where variances and covariances are the strategy parameters. In order to avoid the overly technical details, I will not discuss the most general case of *correlated mutations*, where the full covariance matrix is subject to self-adaptation (see e.g. [10]), but restrict attention to the self-adaptation of $n_\sigma \in \{1, n\}$ variances σ_i^2. For $n_\sigma = n$, each object variable x_i is mutated according to a normally distributed random number $z_i \sim \mathbf{N}(0, \sigma_i'^2)$, i.e.,

$$x_i' = x_i + z_i \quad , \tag{6}$$

and the variance $\sigma_i'^2$ is obtained from mutating σ_i according to a logarithmic normal distribution, i.e.,

$$\sigma_i' = \sigma_i \cdot \exp(s') \cdot \exp(s_i) \quad , \tag{7}$$

where $s' \sim \mathbf{N}(0, \frac{1}{2n})$ and $s_i \sim \mathbf{N}(0, \frac{1}{2\sqrt{n}})$ (s' is identical for the complete individual, while s_i is sampled anew for each component). For $n_\sigma = 1$, the modification of σ reduces to

$$\sigma' = \sigma \cdot \exp(s_0) \tag{8}$$

where $s_0 \sim \mathbf{N}(\frac{1}{n})$. The values $\tau'^2 = \frac{1}{2n}$, $\tau_i^2 = \frac{1}{2\sqrt{n}}$, and $\tau_0^2 = \frac{1}{n}$ of the variances were proposed by Schwefel (see [40], p. 168) and result from partially theoretical investigations.

The choice of a logarithmic normal distribution for the mutation of mutation step sizes σ_i is motivated by just empirical but nevertheless "naturally reasonable" arguments:

---

[2] And, indepedently, of David B. Fogel, who reinvented the method in EP [17].

[3] The notation $x \sim \mathbf{N}(\zeta, \sigma^2)$ indicates x to be a realization of a random variable X which is normally distributed with expectation ζ and variance σ^2.

- A multiplicative modification preserves positive values.
- The median of a multiplicative modification should equal one to guarantee that, on average, a multiplication by a value c occurs with the same probability as a multiplication by $1/c$.
- Small modifications should occur more often than large ones.

Extensive experimental investigations by Schwefel [41, 42, 44] clarified that this mechanism for self-adaptation is extremely robust with respect to the setting of the meta-parameters (or *learning rates*) τ, τ', and τ_0, respectively. Presently, a variety of further work clearly demonstrates that the general principle also works for the adaptation of other parameters such as crossover exchange probabilities [33], mutation rates in canonical genetic algorithms [2], mutation rates in evolutionary programming for the evolution of finite state machines [19], mutation rates for discrete object variables in a hybrid algorithm of ES and GA [9, 39], and momentum adaptation in ESs [34].

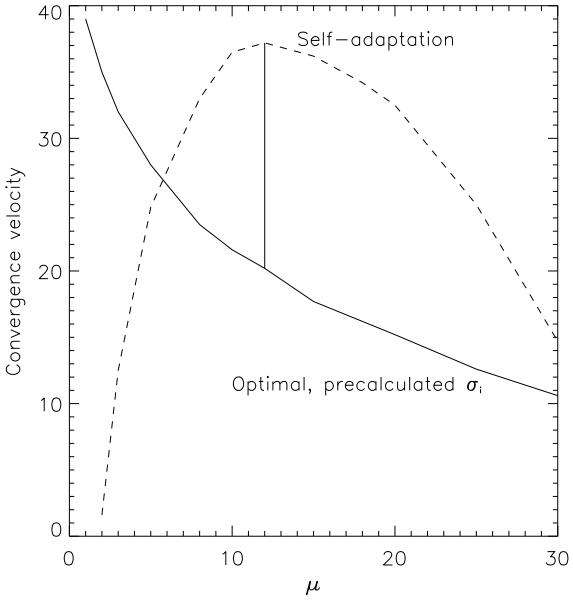

Fig. 5.: Progress rates for the (μ,100)-ES using optimally, precalculated σ_i and the self-adaptive strategy.

To illustrate some of the insights regarding the self-adaptation of $n_\sigma = n$ variances in ESs, I will briefly discuss one of Schwefel's well-designed experiments, using the objective function $F(\mathbf{x}) = \sum_{i=1}^{n} i \cdot x_i^2$ [43]. For this function,

each variable is differently scaled, such that self-adaptation requires to learn the scaling[4] of n different σ_i. Furthermore, the optimal settings of standard deviations $\sigma_i^* \propto 1/\sqrt{i}$ are known in advance, such that self-adaptation can be compared to an ES using optimally adjusted σ_i for mutation. The experiment compares the convergence velocity, measured in terms of a logarithmic scale $\ln f_t'/f_0'$, of (μ,100)-ESs with self-adaptation and the optimal strategy, varying μ within the range $\{1,\ldots,30\}$. The convergence velocity for both strategies is shown in figure 5 as a function of μ.

For the strategy using optimal standard deviations σ_i, the convergence rate is maximal for $\mu = 1$, because this setting exploits the perfect knowledge in an optimal sense. The self-adaptive strategy reaches optimal performance for a value of $\mu = 12$, and both larger and smaller values of μ cause a loss of convergence speed. About 12 imperfect parents collectively yield a performance by means of self-adaptation which almost equals the performance of the perfect (1,100)-strategy and clearly outperforms the collection of 12 perfect individuals.

Clearly, self-adaptation is a mechanism that requires the existence of a knowledge diversity (a diversity of *internal models*), i.e., a number of parents larger than one, and benefits from a collective intelligence phenomenon. Further experiments have shown that the limited life span of individuals (to allow for forgetting misadapted strategy parameters), the creation of a surplus $\lambda > \mu$ of offspring individuals, and the application of recombination also on strategy parameters seem to be important prerequisites for a successful self-adaptation.

Of course, more experimental and especially theoretical work needs to be done in order to understand the self-adaptation process in detail, but the fact that it works in a variety of implementations and for large ranges of parameter settings for the "learning rates" (determining the speed of self-adaptation) provides clear evidence that the basic principle is extremely useful and widely applicable.

3 Completing the Evolution Strategy

After describing the structure of individuals — a vector $\mathbf{x} \in I\!R^n$ of object variables and a vector $\boldsymbol{\sigma}$ of variances (and, for correlated mutations, a vector $\boldsymbol{\alpha}$ of rotation angles representing the covariances; see [4]) — and the principle of self-adaptation, especially regarding the mechanism of the mutation operator, only few remains to complete a standard (μ,λ)-ES algorithm.

Because the population sizes of parent and offspring population are different, the reduction from λ to μ individuals by selection (performed by just determ-

[4] Similarly, Schwefel utilized the sphere model for investigations regarding the self-adaptation of $n_\sigma = 1$ standard deviation, and the function

$$F'(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j^2 \right) \tag{9}$$

for correlated mutations [43].

inistically choosing the μ best offspring as new parents) has to be compensated by a repeated application of either mutation or recombination to create λ individuals. In ESs, this is achieved by recombination, which generates one offspring individual from ϱ ($1 \leq \varrho \leq \mu$) randomly selected parent individuals per application. Typically, $\varrho = 2$ or $\varrho = \mu$ (so-called *global* recombination) are chosen. The λ-fold application of recombination to the parent population yields λ individuals, each of which is mutated and enters the offspring population. The objective function values of the offspring individuals are calculated, selection chooses the μ best to survive, and the recombination-mutation-evaluation-selection cycle is repeated until a termination criterion is fulfilled. The initial population is either generated at random or by means of mutation from a single, predefined starting point.

In contrast to GAs, recombination in ESs is a more flexible operator, incorporating between 2 and μ parents for the creation of a single descendant. Recombination types on object variables and strategy parameters are usually different, and typical examples are *discrete recombination* (random exchanges between parents) for x_i and *intermediary recombination* (arithmetic averaging) for σ_i (see e.g. [10] for details).

That's all to complete the ES — a lot of details have been added, but they are not mandatory for a working strategy (the details are described e.g. in [44], including reasons why they are incroporated into the algorithm). The most recent proposal for a "contemporary ES" incorporates some additional ideas from GAs (such as the recombination probability and multi-point crossover), multirecombination with an arbitrary number ϱ of parents, a life span κ between the two extremes one ((μ,λ)-selection) and infinity ($(\mu+\lambda)$-selection, where the parents are choosen out of offspring *and* old parents), and thus introduces a variety of new possibilities for experimental investigations of evolutionary principles and their effectiveness for the purpose of optimization (see [46] for details regarding the contemporary ES).

4 Theory

Traditional GA-theory, concentrating on the notion of schemata, building blocks, and the schema-theorem giving an upper bound on the expected number of instances of a schema at generation $t + 1$ [21, 25] does not yield any kind of constructive result regarding optimization properties such as global convergence or convergence velocity — and is therefore simply useless for such applications. On the contrary, ES-theory has always focused on these properties, and already Rechenberg calculated optimal step size and maximal convergence velocity of a (1+1)-ES for the corridor and sphere model [35].

Here, I will neither present theoretical results for the simple (1+1)–ES (which does not use a population but just one individual) or any other strategy based on elitist selection, nor discuss any of the mathematical derivations, but I will just provide an overview of the results for (μ,λ)-ESs on the sphere model

$$f(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^*\|^2 = r^2 \quad , \tag{10}$$

where r equals the Euclidean distance between a search point $\mathbf{x}$ and the optimum $\mathbf{x}^*$. For this model, the *relative progress* at generation t is defined by

$$P = \frac{r_t - r_{t+1}}{r_t} \quad , \tag{11}$$

and for $n_\sigma = 1$ and $n \gg 1$, its expectation for a (1,λ)-ES is given by the quadratic equation

$$E[P_{(1,\lambda)}] = \frac{2\hat{\sigma}c_{1,\lambda} - \hat{\sigma}^2}{2n} \quad , \tag{12}$$

using the normalized, dimensionless variable $\hat{\sigma} = \sigma \cdot n/r$. Notice that the equation reflects a *progress gain* $\hat{\sigma}c_{1,\lambda}/n$ as well as a *progress loss* $\hat{\sigma}^2/2n$, and these two terms have to be kept in balance (either by increasing the gain or by decreasing the loss) in order to achieve positive convergence velocity. The constant $c_{1,\lambda}$ denotes the expectation of the maximum of λ stochastically independent standardized normally distributed random variables and reflects the strong relationship between (μ,λ)-selection and the theory of *order statistics* (see [5] for details).

From equation (12), the optimal standard deviation

$$\sigma^* = c_{1,\lambda} \cdot \frac{r}{n} \tag{13}$$

and the resulting maximal relative progress

$$E[P^*_{1,\lambda}] = \frac{c^2_{1,\lambda}}{2n} \tag{14}$$

are easily obtained. The *convergence velocity* φ (or its normalized counterpart $\hat{\varphi} = \varphi \cdot n/r$) is related to the relative progress according to $E[P_{1,\lambda}] = \varphi/r = \hat{\varphi}/n$, such that, using the asymptotic result $c_{1,\lambda} \approx \sqrt{2\ln\lambda}$ [8], one finally obtains

$$\hat{\varphi}^*_{(1,\lambda)} \approx \ln\lambda \quad . \tag{15}$$

For the details of this derivation, the reader is referred to [45].

In case of the (μ,λ)-ES without recombination, the progress coefficients $c_{\mu,\lambda}$ are generalized to reflect the expectation of the average of the μ best offspring, the convergence velocity is given by the equation

$$\hat{\varphi}_{(\mu,\lambda)} = \hat{\sigma}c_{\mu,\lambda} - \frac{1}{2}\hat{\sigma}^2 \quad , \tag{16}$$

and one obtains the results

$$\hat{\sigma}^* = c_{\mu,\lambda} \tag{17}$$

and

$$\hat{\varphi}^*_{(\mu,\lambda)} = \frac{1}{2}c^2_{\mu,\lambda} \quad , \tag{18}$$

which asymptotically yields

$$\hat{\varphi}^*_{(\mu,\lambda)} \sim \ln\frac{\lambda}{\mu} \tag{19}$$

($c_{\mu,\lambda} \approx \mathcal{O}(\sqrt{\ln \lambda/\mu})$; see [11] for details). This result reflects the dominating impact of the *selective pressure* λ/μ for the convergence velocity.

Taking also global recombination (i.e., with $\varrho = \mu$ parents involved in the creation of a single offspring) into account, Beyer and Rechenberg recently derived the equation

$$\hat{\varphi}_{(\mu,\lambda),\varrho=\mu_I} = \hat{\sigma} c_{\mu/\mu,\lambda} - \frac{1}{\mu}\frac{\hat{\sigma}^2}{2} \quad , \tag{20}$$

resulting in the optimal step size

$$\hat{\sigma}^*_I = \mu c_{\mu/\mu,\lambda} \tag{21}$$

and the corresponding convergence velocity

$$\hat{\varphi}^*_{(\mu,\lambda),\varrho=\mu_I} = \frac{\mu}{2} \cdot c^2_{\mu/\mu,\lambda} \tag{22}$$

for intermediary recombination on the sphere model [11, 36]. With the asymptotical result $c_{\mu/\mu,\lambda} \approx \sqrt{\ln \lambda/\mu}$ (the relation $0 \leq c_{\mu/\mu,\lambda} \leq c_{\mu,\lambda} \leq c_{1,\lambda} < \sqrt{2 \ln \lambda}$ holds for the various progress coefficients), a μ-fold speedup of intermediary recombination as opposed to no recombination is obtained:

$$\hat{\varphi}^*_{(\mu,\lambda),\varrho=\mu_I} \approx \mu \ln \frac{\lambda}{\mu} \quad . \tag{23}$$

Surprisingly, for discrete recombination Beyer obtained the same result for the optimal convergence velocity $\hat{\varphi}^*$ from the progress equation

$$\hat{\varphi}^*_{(\mu,\lambda),\varrho=\mu_D} = \sqrt{\mu}\hat{\sigma} c_{\mu/\mu,\lambda} - \frac{1}{2}\hat{\sigma}^2 \quad , \tag{24}$$

but the optimal step size

$$\hat{\sigma}^*_D = \sqrt{\mu} c_{\mu/\mu,\lambda} \tag{25}$$

turns out to be a factor of $\sqrt{\mu}$ smaller.

Resulting from his mathematical analysis, Beyer is able to explain the beneficial effect of recombination by the fact that recombination reduces the progress loss term in the quadratic equation for $\hat{\varphi}$. In other words, recombination serves as a statistical error correction method, reducing the impact of the harmful part of mutations. This effect, called *genetic repair* by Beyer [13], clarifies that the conjecture of the building block hypothesis that the combination of good partial solutions explains the advantage of recombination does not hold for ESs. Also for discrete recombination, Beyer concludes that it works by implicitly performing a genetic repair, such that (by analogy with uniform crossover in GAs [47]), it is very likely that his results will also fundamentally change the direction of further theoretical research regarding GAs (see [12] for a summary).

5 So What ?

The purpose of this article is not to compare ESs and GAs and to claim that the former or the latter are in some sense better or worse — both classes of evolutionary algorithms have their advantages, and both certainly have their preferable domain of application problems. As a rule of thumb, I personally claim that ESs should be applied in case of continuous problems whereas GAs serve most useful in case of pseudoboolean problems. Obviously, hybridizations of both algorithms are promising for the application to mixed-integer problems involving both discrete and continuous object variables (see [9] for an application of such a hybrid to optical filter design problems).

Furthermore, GAs and ESs offer many possibilities for cross-fertilization by testing certain principles of one algorithm within the context of the other. The most promising (and, at the same time, most complicated to understand) potential for cross-fertilization can be identified in the technique of self-adaptation of strategy parameters, which certainly offers a powerful alternative to the common utilization of a constant mutation rate in GAs — in fact, I have shown that for a simple pseudoboolean objective function, the optimal mutation rate critically depends on both search space dimension ℓ and distance to the optimum [3].

Concerning the theoretical understanding of ESs, it is undoubtedly true that, from the very beginning, much theoretical effort was invested to derive results concerning global convergence and convergence velocity, such that presently a relatively deep understanding of their working principles is available. Most recent investigations regarding recombination demonstrate the validity of its interpretation as a genetic repair operator rather than the superposition effect of good partial solutions as claimed by the building block hypothesis of GAs.

GAs are known more widely than ESs, and the number of researchers using GAs for practical applications is much larger than for ESs, which are still basically used nowhere except in Germany. Their usefulness, however, has been clearly demonstrated not only by an extensive comparison with a variety of traditional search methods [44], but also by a large number of practical applications in domains such as biology, chemistry, computer aided design, physics, medicine, production planning, etc. [6].

Acknowledgements

The author gratefully acknowledges financial support by the project EVOALG, grant 01 IB 403 A from the German BMBF. EVOALG is a joint research project of the Informatik Centrum Dortmund (ICD), the Humboldt-Universität zu Berlin, and Siemens AG Munich.

The author would also like to thank Marc Schoenauer for the invitation to present this talk at Evolution Artificielle 1995.

References

1. J. T. Alander. An indexed bibliography of genetic algorithms: Years 1957–1993. Art of CAD Ltd, Espoo, Finland, 1994.
2. Th. Bäck. Self-Adaptation in Genetic Algorithms. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life*, pages 263–271. The MIT Press, Cambridge, MA, 1992.
3. Th. Bäck. Optimal mutation rates in genetic search. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 2–8. Morgan Kaufmann, San Mateo, CA, 1993.
4. Th. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1995.
5. Th. Bäck. Generalized convergence models for tournament- and (μ,λ)-selection. In L. Eshelman, editor, *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 2–8. Morgan Kaufmann Publishers, San Francisco, CA, 1995.
6. Th. Bäck, F. Hoffmeister, and H.-P. Schwefel. Applications of evolutionary algorithms. Report of the Systems Analysis Research Group SYS–2/92, University of Dortmund, Department of Computer Science, February 1992.
7. Th. Bäck and S. Khuri. An evolutionary heuristic for the maximum independent set problem. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 531–535. IEEE Press, 1994.
8. Th. Bäck, G. Rudolph, and H.-P. Schwefel. Evolutionary programming and evolution strategies: Similarities and differences. In D. B. Fogel and W. Atmar, editors, *Proceedings of the Second Annual Conference on Evolutionary Programming*, pages 11–22. Evolutionary Programming Society, San Diego, CA, 1993.
9. Th. Bäck and M. Schütz. Evolution strategies for mixed-integer optimization of optical multilayer systems. In *Proceedings of the 4th Annual Conference on Evolutionary Programming*, 1995.
10. Th. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.
11. H.-G. Beyer. Towards a theory of 'evolution strategies' — Results from the N-dependent (μ,λ) and the multi-recombinant (μ/μ,λ)-theory. Report of the Systems Analysis Research Group SYS–5/94, University of Dortmund, Department of Computer Science, 1994.
12. H.-G. Beyer. How GAs do NOT work. Understanding GAs without Schemata and Building Blocks. Report of the Systems Analysis Research Group SYS–2/95, University of Dortmund, Department of Computer Science, 1995.
13. H.-G. Beyer. Toward a theory of evolution strategies: On the benefits of sex — the $(\mu/\mu, \lambda)$-theory. *Evolutionary Computation*, 3(1):81–111, 1995.
14. R. A. Caruna and J. D. Schaffer. Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In J. Laird, editor, *Proceedings of the 5th International Conference on Machine Learning*, pages 153–161. Morgan Kaufmann Publishers, San Mateo, CA, 1988.
15. L. Davis, editor. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.
16. R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Computer Journal*, 6:163–168, 1963.
17. D. B. Fogel. *Evolving Artificial Intelligence*. PhD thesis, University of California, San Diego, CA, 1992.

18. D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, Piscataway, NJ, 1995.
19. L. Fogel, D. B. Fogel, and P. J. Angeline. A preliminary investigation on extending evolutionary programming to include self-adaptation on finite state machines. *Informatica*, 18:387–398, 1994.
20. L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. Wiley, New York, 1966.
21. D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison Wesley, Reading, MA, 1989.
22. W. Gottschalk. *Allgemeine Genetik*. Georg Thieme Verlag, Stuttgart, 3 edition, 1989.
23. S. Hahn, K. H. Becks, and A. Hemker. Optimizing monte carlo generator parameters using genetic algorithms. In D. Perret-Gallix, editor, *New Computing Techniques in Physics Research II — Proceedings 2nd International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics*, pages 255–265, La Londe-Les-Maures, France, January 13–18 1992. World Scientific, Singapore, 1992.
24. R. Hinterding, H. Gielewski, and T. C. Peachey. On the nature of mutation in genetic algorithms. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the 6th International Conference*, pages 65–72. Morgan Kaufmann Publishers, San Francisco, CA, 1995.
25. J. H. Holland. *Adaptation in natural and artificial systems*. The University of Michigan Press, Ann Arbor, MI, 1975.
26. K. A. De Jong. Are genetic algorithms function optimizers ? In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature 2*, pages 3–13. Elsevier, Amsterdam, 1992.
27. K. A. De Jong. Genetic algorithms are NOT function optimizers. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 5–17. Morgan Kaufmann Publishers, San Mateo, CA, 1993.
28. S. Khuri and Th. Bäck. An evolutionary heuristic for the minimum vertex cover problem. In J. Kunze and H. Stoyan, editors, *KI-94 Workshops (Extended Abstracts)*, pages 83–84. Gesellschaft für Informatik e. V., Bonn, 1994.
29. S. Khuri, Th. Bäck, and J. Heitkötter. An evolutionary approach to combinatorial optimization problems. In D. Cizmar, editor, *Proceedings of the 22nd Annual ACM Computer Science Conference*, pages 66–73. ACM Press, New York, 1994.
30. S. Khuri, Th. Bäck, and J. Heitkötter. The zero/one multiple knapsack problem and genetic algorithms. In E. Deaton, D. Oppenheim, J. Urban, and H. Berghel, editors, *Proceedings of the 1994 ACM Symposium on Applied Computing*, pages 188–193. ACM Press, New York, 1994.
31. E. Mayr. *Toward a new Philosophy of Biology: Observations of an Evolutionist*. The Belknap Press of Harvard University Press, Cambridge, MA, and London, GB, 1988.
32. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, 1994.
33. J. Obalek. Rekombinationsoperatoren für Evolutionsstrategien. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, 1994.
34. A. Ostermeier. An evolution strategy with momentum adaptation of the random number distribution. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature 2*, pages 197–206. Elsevier, Amsterdam, 1992.

35. I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann–Holzboog, Stuttgart, 1973.
36. I. Rechenberg. *Evolutionsstrategie '94*, volume 1 of *Werkstatt Bionik und Evolutionstechnik*. frommann–holzboog, Stuttgart, 1994.
37. G. Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks, Special Issue on Evolutionary Computation*, 5(1):96–101, 1994.
38. G. Rudolph. Convergence of non-elitist strategies. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, *Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 63–66. IEEE Press, 1994.
39. M. Schütz. Eine Evolutionsstrategie für gemischt-ganzzahlige Optimierungsprobleme mit variabler Dimension. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, 1994.
40. H.-P. Schwefel. *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, volume 26 of *Interdisciplinary Systems Research*. Birkhäuser, Basel, 1977.
41. H.-P. Schwefel. Evolutionary learning optimum–seeking on parallel computer architectures. In A. Sydow, S. G. Tzafestas, and R. Vichnevetsky, editors, *Proceedings of the International Symposium on Systems Analysis and Simulation 1988, I: Theory and Foundations*, pages 217–225. Akademie-Verlag, Berlin, September 1988.
42. H.-P. Schwefel. Imitating evolution: Collective, two-level learning processes. In U. Witt, editor, *Explaining Process and Change — Approaches to Evolutionary Economics*, pages 49–63. The University of Michigan Press, Ann Arbor, MI, 1992.
43. H.-P. Schwefel. Natural evolution and collective optimum–seeking. In A. Sydow, editor, *Computational Systems Analysis: Topics and Trends*, pages 5–14. Elsevier, Amsterdam, 1992.
44. H.-P. Schwefel. *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology Series. Wiley, New York, 1995.
45. H.-P. Schwefel and Th. Bäck. Evolution strategies ii: Theoretical aspects. In J. Périaux and G. Winter, editors, *Genetic Algorithms in Engineering and Computer Science*, chapter 7. Wiley, Chichester, 1995.
46. H.-P. Schwefel and G. Rudolph. Contemporary evolution strategies. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, editors, *Advances in Artificial Life. Third International Conference on Artificial Life*, volume 929 of *Lecture Notes in Artificial Intelligence*, pages 893–907. Springer, Berlin, 1995.
47. G. Syswerda. Uniform crossover in genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the 3rd International Conference on Genetic Algorithms*, pages 2–9. Morgan Kaufmann Publishers, San Mateo, CA, 1989.
48. H.-M. Voigt, H. Mühlenbein, and D. Cvetković. Fuzzy recombination for the breeder genetic algorithm. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the 6th International Conference*, pages 104–111. Morgan Kaufmann Publishers, San Francisco, CA, 1995.

This article was processed using the LaTeX macro package with LLNCS style