UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

How to analyse evolutionary algorithms

Hans-Georg Beyer, Hans-Paul Schwefel,
and Ingo Wegener

No. CI-139/02

Technical Report ISSN 1433-3325 August 2002

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence", at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

How to analyse evolutionary algorithms

Hans-Georg Beyer [a,1], Hans-Paul Schwefel [a,2] and
Ingo Wegener [b,2]

[a] *FB Informatik LS 11, Univ. Dortmund, 44221 Dortmund, Germany*
`beyer, schwefel@Ls11.cs.uni-dortmund.de`
[b] *FB Informatik LS 2, Univ. Dortmund, 44221 Dortmund, Germany*
`wegener@Ls2.cs.uni-dortmund.de`

Abstract

Many variants of evolutionary algorithms have been designed and applied. The experimental knowledge is immense. The rigorous analysis of evolutionary algorithms is difficult, but such a theory can help to understand, design, and teach evolutionary algorithms. In this survey, first the history of attempts to analyse evolutionary algorithms is described and then new methods for continuous as well as discrete search spaces are presented and discussed.

1 Some history of evolutionary algorithms

Evolutionary algorithms (EA) form a class of probabilistic optimization methods that are inspired by some presumed principles of organic evolution. Whether such inspiration is helpful or hampering, a neutral side aspect, or an opportunity to build bridges between the islands of different disciplines forming the cluster of human knowledge, may be debated controversially, but not in this contribution. It is simply a matter of fact that EA have become a welcomed tool for tackling the search for extrema, e.g. optimal parameters within simulation models (Schwefel [1]), that withstand classical approaches. Subsequently mentioning only three spatially different though nearly contemporaneous sources (earliest traces go all back to the early 1960s, instead we cite some later but better-known ones)

- evolutionary programming (EP) (Fogel, Owens, and Walsh [2])
- genetic algorithms (GA) (Holland [3])
- evolution strategies (ES) (Rechenberg [4], Schwefel [5])

does not mean that there were not more inventors of the same or at least similar ideas. Fogel [6] has made an attempt to collect a fossil record of the early birds in the field. This field called evolutionary computation (EC) since members of the three teams mentioned above met at conferences like PPSN (Parallel Problem Solving from Nature (Schwefel and Männer [7])), ICGA (International Conference on Genetic Algorithms (Belew and Booker [8])), and EP (Evolutionary Programming (Fogel and Atmar [9])), has got an accommodation in computer science under the roof of computational intelligence (CI) or soft computing or bio-inspired or natural computation together with two other fields, i.e. neural and fuzzy computation. A series of three handbooks (Bäck, Fogel, and Michalewicz [10], Fiesler and Beale

[1] This author is supported as Heisenberg fellow of the DFG under grant Be 1578/4-2.
[2] These authors were supported by the Deutsche Forschungsgesellschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

[11], Ruspini, Bonissone, and Pedrycz [12]) as well as concurrent conferences every four years since 1994 under the umbrella "World Congress on Computational Intelligence" (Michalewicz et al. [13], Fogel et al. [14]) may serve as witnesses of the broad interest this set of methods has gained, recently.

The general frame of EP, GA, and ES is essentially the same and very simply summarized by a loop over partially randomized variation and selection operators steering exploration and exploitation (or chance and necessity) and, in contrast to traditional optimization procedures, acting upon a set of search points in the decision variable space. That is why some of the theoretical investigations mentioned later lead to results that are valid for nearly all simple EA. Nevertheless, due to the different origins, some features of the "canonical" versions of the algorithms are quite specific, and some people still speak of schools or demes that have emphasized or still emphasize their beloved flourish. Therefore, a few remarks seem appropriate about the three kindergartens. To do this we use the popular nomenclature (see Beyer et al. [15]). It should be intuitive enough so that we do not need sophisticated definitions here for an individual (set of variables), its fitness (objective function value), or a generation (one iteration loop with μ parents and their λ offspring) etc.

Evolutionary programming (EP) was first devised to let finite state machines become more and more "intelligent by means of simulated evolution". One or more out of a couple of distinct small manipulations of the state diagram of a parent machine, i.e. a (uniformly distributed random) mutation, offers an offspring. Usually, each parent creates one child. No recombination is applied. Selection takes place as a series of tournaments (the pendant of the proverbial "struggle for life") each with a subset of the contemporary competitors. Those individuals earning highest scores, exactly 50%, enter the next generation. Later, Fogel [16] revised his father's original EP in different ways, some of which resemble more or less the evolution strategies as used in the case of real-valued parameter optimization. Not making use of recombination has remained a "philosophical" distinction to all other EA (see Fogel [17]). We do not discuss this further than mentioning that the evolving entities are thought of as species instead of individuals - and by definition, species do not exchange genetic material/information.

Genetic algorithms (GA) initially served as simplified models of organic evolution in order to investigate adaptation capabilities that might be useful examples for other disciplines, as well. Despite that older members of this school still today emphasize that GA are no optimization methods, it is just that domain where they have earned appreciation including money. The evolving entities are genomes carrying the phenotypic characteristics in coded form, usually making use of an alphabet with low cardinality, on a digital computer consequently in binary form. The initial population is typically generated by drawing all bits with same probability for zeros and ones (or pure random setting within non-binary finite search regions). The main variation operator is recombination, more precisely crossover, e.g. two-point crossover. In this case, the bitstrings of two parents are cut at two random positions and put together by exchanging the innermost parts between the parents, thus creating two offspring at a time. Discussions whether it is better to use both or only one of them, are still ongoing. Not all reproductions underlie recombination (canonically 30% not), so that some individuals are either clones or survivors from the last generation. Mutation, i.e. flipping a bit at this or that position, has been introduced with low probability (e.g. 0.1%) to prevent that a small population loses a still needed one or zero prematurely. In many applications, higher mutation as well as crossover probabilities have become popular, e.g. $1/n$ as mutation probability in case of a genome with n bits and one as crossover probability. Selection takes place when the partners are drawn for recombination. Those who own higher fitness values (in case of minimization of course those with lower objective function values) are preferred. This may be done by ranking the individuals, or canonically, by giving them a chance that is proportional to their (always positive, if necessary transformed) fitness.

Evolution strategies (ES) were devised as experimental optimization techniques, e.g. to drive a flexible device step by step into its optimal state. The first experiments were performed with just one ancestor and one descendant per generation and mutations created by subtracting two numbers drawn from a binomial distribution. The ancestor was replaced by its offspring if the latter was not worse than the former. As soon as computers became available, this two membered or $(1+1)$-ES was accompanied by the multimembered version with recombination. Now, μ parents create λ offspring within one reproduction cycle. Two or even more parents are involved in the recombination step, two extreme forms of which are called discrete (or dominant) and intermediate, respectively. In the case of intermediate recombination, the average of the parental variable values is transferred to the offspring, whereas discrete recombination (like uniform crossover in GA) chooses each component from one of the parents at random. No check is imposed that the parents involved are all different, and there is no mating selection, all parents have the same chance to be chosen. Additionally to 100% recombination, 100% mutation takes place with maximum entropy probability distributions (geometrical for integer variables) or probability densities (normally distributed in case of continuous variables). If the parents for the next generation are drawn from the offspring only – this scheme is called (μ, λ)-ES – there must be a birth surplus, obviously. Otherwise, all parents take place, too, in the $(\mu + \lambda)$-ES, the extreme form of which with $\lambda = 1$ is called "steady state", as has been done with the corresponding GA version. Selection is performed in a strictly deterministic manner and has been called truncation selection, because except for the best μ individuals all others are discarded/forgotten. The so far best individual may be stored outside the population, of course.

Both comma and plus selection schemes are the extremes of a more general $(\mu, \kappa, \lambda, \rho)$-ES with κ as upper limit of the number of reproduction cycles an individual is staying in the population and ρ as the number of parents involved in the recombination step for each offspring. The special notation of a $(\mu/\rho, \lambda)$-ES stands for a comma version with so-called multirecombination, i.e. inheriting to each descendant parameter values that represent the average over ρ parents – the ultimate case being $\rho = \mu$ in one direction and $\rho = 1$ (no recombination) in the other.

Other variants of these three early variants are now collected under the notion of recombinant evolutionary algorithms (EA). Hundreds if not thousands of other incarnations have been proposed and applied. A data base of US patents revealed 67 procedures that bear the name GA in their headline – despite unfinished discussions about when an EA is no longer a GA. For quite a while binary encoding of the decision variables seemed to be a necessary ingredient – until real-coded GA entered the literature, (see, e.g., Eshelman and Schaffer [18]), even with deterministic truncation selection (Mühlenbein and Schlierkamp-Voosen [19]). Due to the fact that probably more than 2000 articles are published annually since a couple of years (see Alander [20]), it is more likely than not that some features of the strategies are reinvented, probably under different names, and same names do not guarantee same features, respectively. Some recently introduced crossover operators produce variations that are traditionally expected under the name mutation.

Until recently, the number of rigorously proven facts about the behavior of EA has been rather small. Nevertheless, there have been some strong beliefs upon which decisions about choosing one or the other version have been taken. Some of them turned out to be wrong, others are still unproven hypotheses or summaries of empirical experience. Repeating arguments and counter-arguments from finished or still ongoing discussions would fill too many pages and turn out as boring for the uninitiated. That is why we restrict our report to only some, maybe called central, discussions of the past and then turn to the presence, especially to most recent hard facts.

First analyses of the ES performance concentrated on the so-called progress velocity, i.e. the average distance in the search space traveled in the useful direction per function evaluation. This local measure was considered for the two-membered ES with uniform random discrete mutations in the Moore neighborhood of the parent on an inclined plane, a parabolic ridge, and a parabolic top with circular level lines. The useful direction in case of the inclined plane was the gradient direction, in case of the ridge the straight line connecting the vertices of the parabolic level lines, and in case of the top any reduction of the distance to the summit was considered as useful. Schwefel [21] observed that such discrete mutations can lead to stagnation of the search somewhere on the ridge and to a considerable decrease of the progress velocity when approaching the hilltop. He proposed to use more versatile variation schemes with smaller as well as larger mutations, e.g., according to a Gaussian probability density distribution with zero mean and given standard deviation for each (continuous) variable. For such continuous mutations Rechenberg [4] found asymptotic approximations of the progress velocity of a two-membered ES on two model functions, a spherical model as the parabolic top above and a corridor model, which resembles an n-dimensional rectangular ridge. In both cases the progress rate (expected distance traveled per objective function call) only depends on the number of variables, the standard deviation of the mutations (same for all directions), and a topology parameter, i.e., the distance from the optimum in case of the hypersphere or the corridor width (same for $n-1$ perpendicular directions in the n-dimensional space) in case of the rectangular ridge. Dividing the progress velocity and the standard deviation by the topology parameter and multiplying both items with the number of variables, the formulas become simple relations between the normalized progress rate and the normalized "mean step size" or square root of the single mutation variance. This relation has a maximum that in both cases corresponds to a success probability (the probability of replacing the parent by the offspring) in the vicinity of 20%. If the standard deviation is smaller than at this maximum, then the success probability is higher, but the search is slower; if, however, the mean step size is larger than optimal, both the progress rate and the success probability decline until they vanish at infinitely large mutations. At least 50% of the maximal progress rate can be achieved within an "evolution window", a range of about one decade concerning values of the standard deviation.

The monotonicity of the success probability over the mutation strength has led to a simple rule for adjusting the latter (1/5 success rule). This investigation was extended by Schwefel [5, 22] for multimembered ES with λ descendants per generation and just one parent, thus necessarily without recombination. Both the comma and the plus versions were considered. The asymptotic approximations of the "universal" laws for normalized progress velocity over normalized standard deviation are of same type as above for all plus versions including $\lambda = 1$, but they differ substantially in case of the comma ES when the standard deviations exceed their optimal values by far. Negative progress rates indicate divergence of the optimum-seeking process when the mutation steps become too large. The maxima of the progress-rate curves increase sublinearly with the number of descendants per generation and differ vanishingly between plus and comma strategies.

First empirical results about a positive influence of recombination on the expected progress velocity of a $(\mu + 1)$-ES were obtained by Rechenberg [4] already. Thus it is wondrous that more often than not people argue recombination to be a secondary variation operator in ES (in contrast to GA, where mutations really were thought to be of secondary importance for a long time).

Self-adaptation of the mutation strength(s) has been considered as of utmost importance from the very beginning of the ES history. Such feature is an ingredient of all classical optimization procedures. Whereas step size control in that domain relies on a more or less sophisticated internal model of the (local) response surface (fitness landscape, otherwise) and a rational processing of the information usually gathered over a series of iterations, a self-adaptive ES would have to consider the objective function as a black box and to operate on less knowledge about its historical pathway (in case of mostly haploid individuals with just one set of genes).

Early empirical investigations (Schwefel [5, 22]) led to the belief that under certain conditions such self-adaptation without exogenous control can be achieved, but not under the $(\mu+1)$- or steady state scheme, because decreasing the mutation strength is always rewarded via an increased success rate. The so-called mutative step-size control operates with individuals that are not only characterized by their vector of object variables, but additionally by one standard deviation used for creating the offspring or even more strategy parameters controlling mutations with more general normal probability density distributions. A birth surplus seems indispensable in order to give the optimal mutation step size a chance to succeed within just one generation. This led to proposing ES with $\lambda > 1$, more generally with as many descendants as are necessary to allow at least one descendant per parent that improves the objective function. Calling the ratio λ/μ birth surplus or selection pressure, this ratio would have to be equal to or higher than the inverse success probability corresponding to the optimal mutation strength with maximal progress velocity. Even up to n different step sizes for the n variables could be envisaged under such premise – if μ was not too small (Schwefel [23]). Dreams of incorporating even more degrees of freedom of the normal distribution by introducing the full correlation matrix with up to $n(n-1)/2$ non-zero correlation coefficients could not be realized at that time to full extent due to a lack of computation power. Rudolph [24] conjectured that $\Omega(n^2)$ individuals in an ES population might be necessary in order to adapt so many strategic parameters representing the "internal models" of the individuals' environment.

Despite of enduring controversial discussions, Holland's schema theorem [3] is still a corner stone of the GA theory. A schema is a bitstring with one or more don't care symbols "*" and thus represents 2^d different bitstrings with d as number of the "*". Holland expressed the expected number of offspring representing some schema after applying proportional selection, one-point crossover, and mutation in terms of an inequality with the number of parents belonging to the same schema on the right hand multiplied by three factors. The first factor is the average fitness of the parental schema divided by the average of the whole population; this factor is thus greater than one for above average parents (on the premise of diversity among the parents). Both other factors are less than one and represent probabilities of harmful recombinations and harmful mutations. The first factor has been rewritten as $1+c$, and by assuming c to be a constant over several generations this has led to the belief of an exponential increase of the number of above average fit parental schemata. But c must vanish in approaching an optimum, and the influence of the other factors, being detrimental, finally dominates if the mutation and recombination probabilities don't vanish. Rudolph [25] found that a canonical (non-elitist) GA finally fluctuates at a certain distance of the optimum, because the best positions get lost again and again. This corresponds, by the way, to the continuous Fisher-Eigen model and its findings (see Maynard Smith [26]). Neglecting improvements by mutation and recombination, the schema theorem does not help in modeling the progress velocity in terms of the so far best solution within a finite population.

Another strong belief concerning GA is the so-called building block hypothesis (BBH, see Goldberg [27]). It states that recombination, e.g. one-point crossover, often enables to put together good parts of one parental bitstring with good other parts of the second parent delivering an even better combination of both in an offspring. Such argument resembles in some way the situation in continuous search spaces where improving steps in several independent directions can be superimposed with overall positive effect. But, this happens only if the objective function is decomposable in some way and the corresponding n independent directions can be found. Generally, such decomposable objective functions are rarely given, and if so, n one-dimensional line searches suffice for finding the optimum. For a more detailed discussion see Grefenstette [28] and Rudolph [25].

Finally, we can ask whether we really need EA, whether EA need features of organic evolution, or not. The second question may be answered by the infamous "yes and no": – No, because any idea improving an algorithm to solve a given problem is feasible, may it resemble biological prototypes or not. The best way to handle a given problem would be the invention of a special method, even a best one if it exists. Its goodness depends merely on our knowledge or ignorance of the problem's characteristics. – Yes, because otherwise the name of the method should be changed – or it becomes deceptive. At least some researchers (like Holland) insist that EA are an instrument to learn about natural processes. The first, even broader question presumably does not lead to an answer which could be agreed upon by all people. Again, one might call for special methods for special problems. But, not willing to spend enough time to invent such special methods, practitioners are cast toward using existing methods even if they are not optimal.

In the following two sections we present new methods how to analyse evolutionary algorithms on continuous (Section 2) and discrete (Section 3) search spaces.

2 New methods for continuous search spaces

It is common belief that evolutionary optimization of real-valued objective functions in $\mathbb{R}^n$ search spaces is a specialty of evolution strategies (ES). While there are indeed state-of-the-art ES versions specially tailored for $\mathbb{R}^n$ supporting this belief, it is historically not correct (for the history see Beyer and Schwefel [29]). The appearance of special ES versions for search in $\mathbb{R}^n$ may be regarded as a consequence of the theory: theoretical investigations on the behavior of EA in $\mathbb{R}^n$ search spaces have been done mainly in the field of ES. As to the other EA, there are only a few exceptions. Concerning real-coded GA, the work of Qi and Palmieri [30] should be mentioned here, where the effect of adaptive (real-valued) mutations on the convergence properties in a GA using fitness-proportional selection has been investigated. Only recently Beyer and Deb [31] started first investigations on the (self-) adaptive behavior of real-coded GA populations and pointed out similarities concerning the convergence order of real-coded GA and ES.

In the early phase of ES, these EA were mainly developed and analyzed by engineers. A more or less system-theoretic approach aiming at the prediction of the EA's behavior as a dynamical system served as the central paradigm. That is, the usual way of thinking about a theory of EA is considering the EA and the objective function $f : \mathbb{R}^n \mapsto \mathbb{R}$ (function to be optimized, often referred to as fitness function) in terms of a dynamical (or evolutionary) system, the "EA system". The goal of this type of theory is therefore to model the real EA system and to predict certain aspects of its behavior.

Evolution strategies as a special version of EA operate on a population of μ parent individuals $\mathfrak{P} = (\mathfrak{a}_1, \ldots, \mathfrak{a}_\mu)$. In general, each individual $\mathfrak{a}_m$ comprises a set of object parameters $\mathbf{y} \in \mathbb{R}^n$ (i.e., the search space variables to be optimized), a secondary set of so-called (endogenous) strategy parameters $\mathbf{s}$, and its fitness function value $f(\mathbf{y})$: $\mathfrak{a}_m = (\mathbf{y}_m, \mathbf{s}_m, f(\mathbf{y}_m))$. By producing λ offspring $\tilde{\mathfrak{a}}_l$ from the parental population $\mathfrak{P}$ via recombination and mutation an offspring population $\tilde{\mathfrak{P}}$ is formed. After that, truncation selection (sometimes called "breeding selection") is applied resulting in a new population forming the parent population at time step (or generation) $t+1$. Depending on whether selection takes only $\tilde{\mathfrak{P}}$ into account or both parent and offspring population $(\mathfrak{P}, \tilde{\mathfrak{P}})$, one speaks of comma selection (denoted by (μ, λ)) and plus selection (denoted by $(\mu + \lambda)$), respectively. The latter case is an elitist selection scheme because it conserves the best individual (with respect to its measured fitness) found so far.

From a formal point of view, the state of the EA at time t is fully determined by the state of the parent population $\mathfrak{P}^{(t)}$. If we include all information which influences the future in the strategy parameters, the stochastic process describing the EA is a memory-less process (or first-order Markov process) whose transition operator will be called $\mathsf{M}^{(t)}$. Let $p^{(t)}(\mathfrak{P})$ be the state density at time step t. Then

$$p^{(t+1)}(\mathfrak{P}) \quad = \quad \mathsf{M}^{(t)} \cdot p^{(t)}(\mathfrak{P}).$$

While this equality describes the dynamics of the EA system completely, its usefulness is rather limited: the analytical determination of the dynamics is almost always excluded. Even in the simplest cases, the analytical determination of the Markov kernel is excluded. Furthermore, the information provided by the $p^{(t)}(\mathfrak{P})$ dynamics is rather difficult to interpret. Spears [32] reports about similar problems during the analysis of EAs on discrete search spaces. One way to circumvent these problems is to investigate infinite instead of finite populations (see Vose [33]). We analyse the orginal process and are satisfied with less universal parameters than the Markov kernel.

Aggregated quantities, especially expected values which can be derived from $p^{(t)}(\mathfrak{P})$, related to the optimization performance are of special interest. When thinking of EA practice, the user often monitors the dynamics of the fitness values, e.g., expected average population fitness and expected best-so-far fitness come into mind. From a theoretical viewpoint also the expected distance $R^{(t)}$ to the optimum state (if there is a single one) is of interest. It should be the aim of theory to predict these mean-value dynamics for a given EA system analytically. However, up until now, even this task can only be accomplished for the simplest EA systems using asymptotic ($n \to \infty$) considerations or by relying on approximations. Later we will report about such analyses using simple fitness functions such as the sphere model and the ridge family.

Before that we investigate some alternatives for characterizing performance aspects of the EA system bypassing the problems with the EA dynamics:

- global convergence proofs,
- order of convergence,
- local performance measures, and
- global performance measures.

Since EA are randomized algorithms, there is always a certain probability of not reaching the optimum state $\hat{\mathbf{y}}$ or a certain vicinity of the optimum (in continuous search spaces) within a finite number of time steps. Therefore, global convergence statements concern the infinite time behavior of the EA. Investigating the convergence of the fitness values $f \in \mathbb{R}$ to the global optimum $\hat{f} := f(\hat{\mathbf{y}})$, one has to show that

$$\Pr\left(|f(\mathbf{y}_{1;\mu}^{(t)}) - \hat{f}| \leq \delta\right)$$

converges (with $t \to \infty$) to 1 for each positive constant δ. Here $\mathbf{y}_{1;\mu}^{(t)}$ represents the best out of the μ parent individuals at time t. The first result of this type, namly for the $(1+1)$-ES with constant Gaussian mutation strength, was sketched by (Rechenberg [4]); a rigorous proof can be found in (Born [34]). Concerning *non*elitist selection schemes, proving or disproving global convergence depends also on the fitness function and the mutation (control) rules. For example, canonical GA, using nonelitist selection schemes like proportionate or tournament selection, are not globally convergent. This aspect has been pointed out by De Jong [35, 36] and formalized and generalized by Rudolph [37, 25]. Davis and Principe [38] have considered the convergence of the population density toward a steady-state density.

Global convergence is often regarded by theoreticians as a minimal prerequisite an EA should obey in order to qualify as a suitable optimization algorithm. Of course, global convergence is (trivially) necessary for locating the optimum with probability one and deriving the expected running time of such algorithms. However, in practice, EA are very often used for evolving approximate solutions under hard cpu-time restrictions, not necessarily the optimal solutions. Therefore, the EA should rather be regarded as amelioration techniques and not as optimization algorithms. Furthermore, it is often desirable to evolve rather robust solutions than to locate a singular peak. All these tasks are not necessarily better served by a globally convergent EA, it might be the case that just the nonconvergent EA versions, e.g. using nonelitist selection techniques, prove better suited for such purposes.

In order to summarize this discussion, proving global convergence is of certain mathematical interest, but it provides a characterization of the EA dynamics much too crude. For example, it does not answer the question how fast the optimum is approached. In the theory of (deterministic local) optimization, the concept of convergence order is used to provide bounds on the dynamics. Rappl [39, 40] was the first to introduce this concept in order to characterize random search methods similar to the (1+1)-ES. One possibility is to consider the dynamics of the expected fitness value toward the optimum $\hat{f}$ (minimization considered here), i.e. $\mathrm{E}(f(\mathbf{y}^{(t)}) - \hat{f})$. He was able to show under certain conditions on the fitness function and a mutation operator with time-constant mutation density that

$$\mathrm{E}(f(\mathbf{y}^{(t)}) - \hat{f}) \quad = \quad t^{-\Theta(1/n)}.$$

This implies an exponential running time. Using a constant mutation density throughout the whole evolution does not yield an efficient algorithm for the problem class considered. Therefore, in continuous search spaces the mutation density should be controlled during evolution. Using sphere-symmetrical mutation densities together with a suitable step-size control for the length of the mutations Rappl [39] was able to prove linear convergence for a class of functions with positive definite Hessian matrix. Linear convergence in mean is obtained if there exists a $q > 1$ such that $q^t \mathrm{E}(f(\mathbf{y}^{(t)}) - \hat{f}) \xrightarrow{t\to\infty} 0$, thus leading to an exponentially fast approach toward the optimum value $\hat{f}$. Interestingly, under the conditions made, it can also be shown that there exists a $\tilde{q} > q > 1$ such that $\tilde{q}^t \mathrm{E}(f(\mathbf{y}^{(t)}) - \hat{f}) \xrightarrow{t\to\infty} \infty$, that is, the dynamics is also bounded from below by an exponential function. Therefore,

$$\mathrm{E}(f(\mathbf{y}^{(t)}) - \hat{f}) = 2^{-\Theta(t)},$$

where the constants in the Θ-expression depend on f and n. This result, which basically holds for convex fitness level sets (with positive definite Hessian), is based on the assumption that the EA "is able" to control the mutation strength (i.e., the expected step size) such that the conditions for the proofs are fulfilled. The mutation control part of the EA is usually not analyzed. The inclusion of the mutation control part in the analysis appears in all cases investigated until now as a difficult task. As for the convergence order analysis the only proof given so far concerns a $(1 + 1)$-ES with success dependent step-size control rule where the step size is increased after a successful mutation by a factor $\gamma_1 > 1$ and decreased by a factor $\gamma_2 \in (0, 1)$ otherwise. The proof of linear convergence in Rappl [39] bears witness to that.

Characterizing EA by their convergence order on specific objective function classes may be regarded as a first step toward a quantitative assessment of the EA's behavior. If an EA system obeys linear convergence order, then we know that the logarithmic expected fitness progress can be bracketed by two linear falling curves of the generation time t. Since evolutionary optimization is performed very often as an online procedure applied to a black box, monitoring the logarithmic fitness dynamics can yield valuable information about the problem behind the black box. However, as practice shows, linear convergence order as such does not necessarily say something about the performance and the computer resources needed in order to reach a certain vicinity of the optimum:

- different EA can have different slopes,
- the computer resources needed for a one-generation time step (basically the number of f-evaluations) can differ for different EA,
- the slope itself depends on f and the dimension of the search space, and
- the EA system may not converge to the optimum (e.g., if the fitness value is disturbed by noise with constant variance, see below).

For this reason, performance measures are needed which evaluate the EA with respect to its local performance (i.e., for one time step) and its global performance (i.e., for a larger number of generations).

Local performance measures (or more generally, progress measures) are expected values of (aggregated) population states. They are usually defined problem-specifically such that they can be used to evaluate the amelioration power of the EA from generation t to generation $t+1$ given the population state $\mathfrak{P}^{(t)}$. The measures in the search space $\mathbb{R}^n$ are called *progress rate* φ and those for the fitness space are called *quality gain* $\overline{Q}$. The latter is defined as

$$\overline{Q} \quad := \quad \mathrm{E}(f(\tilde{\mathbf{y}}^{(t)}) - f(\tilde{\mathbf{y}}^{(t+1)}) | \mathfrak{P}^{(t)}),$$

it measures the expected fitness gain from one generation to the next. The $\tilde{\mathbf{y}}$-vectors are aggregated vectors from the parental population such as the vector $\tilde{\mathbf{y}} = \mathbf{y}_{1;\mu}$ belonging to the best fitness value or the parental population centroid $\tilde{\mathbf{y}} = \langle \mathbf{y} \rangle$.

The progress rate φ measures the expected distance change with respect to a predefined goal $\check{\mathbf{y}}$, i.e.,

$$\varphi \quad := \quad \mathrm{E}(\|\tilde{\mathbf{y}}^{(t)} - \check{\mathbf{y}}\| - \|\tilde{\mathbf{y}}^{(t+1)} - \check{\mathbf{y}}\| \, | \mathfrak{P}^{(t)}),$$

where $\check{\mathbf{y}} = \hat{\mathbf{y}}$ (i.e., the optimum state) is usually used.

From the definitions it becomes clear that – theoretically – these quantities can be used to reconstruct the mean value dynamics of the f-values and the residual distance dynamics, respectively, measuring the approach toward the optimum (provided that $\mathfrak{P}^{(t)}$ is known). However, as one might expect, calculating these local progress quantities is – again – almost always excluded. But, there are some exceptions where the calculations are tractable by the use of approximations or asymptotic techniques. We will discuss some results and derivation ideas below.

Global performance measures are designed for evaluating the long-term behavior of the EA. Here, mainly the aspects of computer resources used are of interest. The expected running time T needed for reaching the optimum or (in continuous search spaces) for reaching a certain vicinity of the optimum is considered. Since the fitness evaluations are usually that part of the EA which is the most time consuming one, efficiency can be measured by counting the number of function evaluations. Of course, having the evolution dynamics at hand, calculating the efficiency is trivial (for an example, see below). However, there are also possibilities to bypass the dynamics. As already discussed, convergence order results can be helpful for providing rough estimates on the expected running time. But it should also be stated that knowing the linear convergence behavior of an EA system does not necessarily imply that the EA has a guaranteed polynomial time complexity of small order. For such results we have to take into account the dependence on f and n.

After these general considerations we will discuss the methods for the analysis of ES using two specific examples. First, we give a short review on the main results obtained from the performance analysis of the $(\mu/\mu_I, \lambda)$-ES on the noisy quadratic sphere. The index I indicates the type of intermediate multirecombination used in this ES. This recombination simply calculates the centroid of the parental population (of size μ). On top of that the λ offspring are generated by adding isotropic Gaussian mutation vectors $\mathbf{z} \sim \sigma\mathcal{N}(\mathbf{0}, \mathbf{1})$ to the parental centroid. The noisy quadratic sphere is defined by

$$f_{\mathrm{ns}}(\mathbf{y}) := \|\mathbf{y}\|^2 + \epsilon, \quad \text{with} \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2).$$

While for $\sigma_\epsilon = 0$ optimizing f_{ns} is one of the simplest tasks in numerical optimization – using a gradient strategy, one needs $n + 3$ function evaluations in order to locate the optimum exactly (within the numerical accuracy) – noisy fitness values deteriorate the performance of most of the deterministic optimization algorithms (for an empirical study, see Arnold and Beyer [41]). Optimization in noisy environments seems therefore an application domain where EA might excel.

Local progress measures are especially useful in cases where the local progress can be used to determine the expected EA system state at $t + 1$ given the state at t. When considering only one progress measure, this implies that the EA's system state must be describable by only one (aggregated) state quantity which – of course – should be related to an observable performance quantity such as the expected fitness or the residual distance to the optimum. The sphere model fulfills this condition perfectly. The system's state can be described by the residual distance r of the parental centroid to the optimum.

Even though the sphere model is highly symmetric and the $(\mu/\mu_I, \lambda)$-ES with isotropic mutations is considered, the calculation of the progress rate and the quality gain, given the parental centroid distance r and the mutation strength σ, cannot be done analytically. However, it is possible to derive asymptotically exact expected values for $n \to \infty$. The basic ideas behind the derivation consist of:

- the decomposition of the $\mathbf{z}$ mutation vector into a gain part x pointing locally in optimum direction $\mathbf{e}_r$ (radial component) and a perpendicular part $\mathbf{h}$ (transversal component)

$$\mathbf{z} = -x\mathbf{e}_r + \mathbf{h}, \quad \text{with} \quad \mathbf{e}_r^{\mathrm{T}}\mathbf{h} = 0,$$

- the introduction of normalized quantities

$$\sigma^* := \sigma\frac{n}{r}, \quad \varphi^* := \varphi\frac{n}{r}, \quad \overline{Q}^* := \overline{Q}\frac{n}{2r^2}, \quad \text{and} \quad \sigma_\epsilon^* := \sigma_\epsilon\frac{n}{2r^2},$$

- identifying random variates in the expected value expressions such that for $n \to \infty$ these quantities become asymptotically normally distributed, e.g., for the quality gain one obtains

$$\Delta_Q^* = \sigma^*\langle x\rangle - \frac{\sigma^{*2}}{2N}\|\langle\mathbf{h}\rangle\|^2,$$

– and the calculation of expected values (e.g., for Δ_Q^*) by the technique of noisy or induced order statistics (see Arnold and Beyer [42] or Beyer [43]).

As a result one obtains

$$\varphi^* \simeq \overline{Q}^* \simeq \sigma^{*2} \left[\frac{c_{\mu/\mu,\lambda}}{\sqrt{\sigma^{*2} + \sigma_\epsilon^{*2}}} - \frac{1}{2\mu} \right],$$

where $c_{\mu/\mu,\lambda}$ is the expected value of the average of the top μ order statistics from the standardized normal variate, the so-called progress coefficient. It can be approximated using the inverse error function

$$c_{\mu/\mu,\lambda} \simeq \frac{\lambda}{\mu} \frac{1}{\sqrt{2\pi}} \exp \left[- \left(\operatorname{erf}^{-1} \left(1 - 2\frac{\mu}{\lambda} \right) \right)^2 \right]$$

where

$$\operatorname{erf}(x) := \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

(asymptotically exact for $\lambda, \mu \to \infty$ with $\mu/\lambda \in (0, 1]$). As a first observation we immediately learn from the result on φ^* that the progress depends on the mutation strength σ^* and the noise σ_ϵ^*. The progress can be even negative indicating (local) divergence (keeping μ and λ as exogenous strategy parameters fixed). The actual dynamics depends on the dynamics of the mutation strength. We will investigate the dynamical behavior of the ES later. Here, we will only discuss qualitative convergence aspects. Non-divergence is ensured as long as $\varphi^* \geq 0$; this leads immediately to the *evolution criterion*

$$\sigma^{*2} + \sigma_\epsilon^{*2} \leq 4\mu^2 c_{\mu/\mu,\lambda}^2.$$

This inequality allows a characterization of the evolutionary amelioration process without knowing the actual dynamics.

Consider the special case of vanishing fitness noise, i.e., $\sigma_\epsilon^* = 0$. We obtain from the evolution criterion $\sigma^* \leq 2\mu c_{\mu/\mu,\lambda}$. What happens under the condition of constant mutation strength σ (i.e., the non-normalized one)? Taking the normalization of σ^* into account we see that σ^* increases with decreasing $r^{(t)}$ (local convergence). However, σ^* can only increase up to the point where the evolution criterion is violated, i.e., $\sigma^* = 2\mu c_{\mu/\mu,\lambda}$ (otherwise $\varphi^* < 0$). That is, we have $\sigma n/r = 2\mu c_{\mu/\mu,\lambda}$ as equilibrium condition and the ES does not converge to the optimum. Instead, the evolution stagnates at a specific r-value, the residual localization error R_∞

$$\sigma = \text{const.} \quad \text{and} \quad \sigma_\epsilon^* = 0 \quad \Longrightarrow \quad R \geq R_\infty = \frac{\sigma n}{2\mu c_{\mu/\mu,\lambda}}.$$

This shows the necessity of controlling σ in comma-strategies in order to approach the optimum arbitrarily close.

A similar behavior can be observed in the case of constant fitness noise σ_ϵ. The corresponding inequality $\sigma_\epsilon^* \leq 2\mu c_{\mu/\mu,\lambda}$ is directly obtained from the evolution criterion above. Using the normalization equations one gets as final localization error bound

$$\sigma_\epsilon = \text{const.} \quad \text{and} \quad \sigma^* = 0 \quad \Longrightarrow \quad R^2 \geq \tilde{R}_\infty^2 = \frac{\sigma_\epsilon n}{4\mu c_{\mu/\mu,\lambda}}.$$

This is an interesting result indicating that an ES system with fixed μ and λ evolving in a fitness landscape under constant fitness noise cannot ameliorate with arbitrary precision – no matter how the mutation strength σ is chosen. That is, such a system cannot be an optimizer in a classical sense. Even though we have considered the quadratic sphere model here, the effect can be observed qualitatively in all EA systems with fixed μ and λ and constant fitness noise (including GA, see Beyer and Arnold [44]).

When considering the results on φ^* one notices that progress toward the optimum is a result of two opposite tendencies: a positive gain part and a negative loss part. The main effect of recombination is due to the reduction of the loss part by a factor of $1/\mu$ compared to the $(1,\lambda)$-ES. The reason for this loss reduction can be traced back to the length-reducing effect when averaging the uncorrelated $\mathbf{h}$ components of the $\mathbf{z}$ mutations by the intermediate recombination. This effect has been coined "genetic repair" (for a detailed explanation see Beyer [45]). While the length of the loss part is reduced, the radial components $x_{m;\lambda}$ of the mutations are only slightly affected: due to the effect of (μ,λ) selection these components are correlated with a tendency pointing into the local improvement direction. Interpreting these observations, one can state that intermediate recombination mainly extracts the similarities from the parents.

Similarity extraction as such does not guarantee a performance increase independent of the fitness function and mutation strength control rule used. Furthermore, one has also to differentiate between a performance increase on the generational level and the efficiency level. For example, in the case of vanishing fitness noise $\sigma_\epsilon^* = 0$ one obtains from the result on φ^* for the maximal progress per generation $\max_{\sigma*} \varphi^*(\sigma^*) \simeq \mu c^2_{\mu/\mu,\lambda}/2$. As can be proven (Beyer [43]) this theoretically maximal generation progress is asymptotically equal to λ times the maximal progress of the $(1+1)$-ES, thus, providing a λ-fold generational speed-up. However, when considering the serial running time, this is brought at the prize of a λ-fold number of function evaluations, i.e., the time for completing a generation is increased by a factor of λ. Defining the *efficiency* η as the normalized progress per fitness evaluation

$$\eta := \frac{\varphi^*}{\lambda},$$

one finds that the efficiency (i.e., the serial performance) of the $(\mu/\mu_I,\lambda)$-ES can be at most that of the $(1+1)$-ES in the noise-free case. That is, using a $(\mu/\mu_I,\lambda)$-ES on the sphere model is of no use. However, one can also show (Arnold and Beyer [42, 46]) that things change positively when considering the noisy case ($\sigma_\epsilon^* > 0$): For sufficiently large noise strengths the efficiency of the $(\mu/\mu_I,\lambda)$-ES exceeds that of the $(1+1)$-ES. Here we have found a first situation where a recombining population in $\mathbb{R}^n$ really can help.

Due to the spherical symmetry of the model considered, the dynamics of the EA system can be characterized by the expected value $r^{(t)}$ of the distance of the parental centroid to the optimum $\hat{\mathbf{y}}$. From the definition of the progress rate and the normalization, one gets $r^{(t)} = r^{(t-1)}(1 - \varphi^*(\sigma^{*(t-1)})/n)$. Since, in general, $\sigma^{*(t-1)}$ is a random variate, $r^{(t)}$ itself is still a conditional expected value. Taking the expectation with respect to $\sigma^{*(t-1)}$ one obtains $\overline{r}^{(t)} = \overline{r}^{(t-1)}(1 - \overline{\varphi^*(\sigma^{*(t-1)})}/n) =: R^{(t)}$ (using R to symbolize the unconditioned expected value). Iteration yields formally

$$R^{(t)} = \prod_{g=0}^{t-1} R^{(0)} \left(1 - \frac{\overline{\varphi^*(\sigma^{*(g)})}}{n}\right),$$

where $\sigma^{*(g)}$ is the time-discrete dynamics of the normalized mutation strength. Using the inequality $\ln(1-x) \le -x$ one can bound the $R^{(t)}$-dynamics by

$$R^{(t)} = R^{(0)} \exp\left[\sum_{g=0}^{t-1} \ln\left(1 - \frac{\overline{\varphi^*(\sigma^{*(g)})}}{n}\right)\right] \le R^{(0)} \exp\left[-\sum_{g=0}^{t-1} \frac{\overline{\varphi^*(\sigma^{*(g)})}}{n}\right].$$

As one can see, the $R^{(t)}$-dynamics is governed by the dynamics of the (normalized) mutation strength, the calculation of which has been managed up until now only for the $(1,\lambda)$-self-adaptive ES (Beyer [47] and recently Arnold [48] for the cumulative step-size adaptation of Hansen and Ostermeier [49, 50]). Without going into the detail here, the main results of these analyses show that both σ-adaptation techniques are able to approach a steady-state behavior (provided that the respective evolution criteria are not violated) with positive expected φ^*. Similar observations have been made by simulations using more complicated fitness functions and different ES versions (see e.g. Schwefel [23]).

In order to proceed with the discussion, we now assume that there is a steady-state $\overline{\varphi}^*$ such that $0 < \overline{\varphi}^* = \text{const.} < \infty$. Taking the logarithm in the equality for $R^(t)$ we find

$$\ln(R^{(t)}) = \ln(R^{(0)}) - t \ln\left(\frac{1}{1 - \overline{\varphi}^*/n}\right).$$

One observes linear convergence in the logarithmic picture, i.e., the ES exhibits linear convergence order. Considering the asymptotic limit $n \to \infty$ one finds the dynamics

$$R^{(t)} \simeq R^{(0)} \exp\left(-\frac{\overline{\varphi^*}}{n} t\right),$$

showing that the residual distance to the optimum reduces exponentially fast.

The inequality on the $R^{(t)}$-dynamics can also be used for estimating the number of generations needed in order to reach a certain vicinity of the optimum. To this end, we assume that the σ-adaptation technique is able to ensure $\overline{\varphi}^* > \check{\varphi}^* > 0$ after a certain time period t_0. Using this $\check{\varphi}^*$, we obtain

$$\begin{aligned} R^{(T+t_0)} &\le R^{(t_0)} \exp\left[-\sum_{g=t_0}^{t_0+T-1} \frac{\overline{\varphi^*(\sigma^{*(g)})}}{n}\right] \\ &\le R^{(t_0)} \exp\left[-\sum_{g=t_0}^{t_0+T-1} \frac{\check{\varphi}^*}{n}\right] = R^{(t_0)} \exp\left[-\frac{\check{\varphi}^*}{n} T\right]. \end{aligned}$$

Resolving for t, one finds

$$T(n) \le \frac{n}{\check{\varphi}^*} \ln\left(\frac{R^{(t_0)}}{R^{(t_0+T)}}\right)$$

and the number ν of function evaluations can be bounded by

$$\nu(n) \le \frac{n}{\check{\eta}} \ln\left(\frac{R^{(t_0)}}{R^{(t_0+T)}}\right),$$

where $\check{\eta} = \check{\varphi}^*/\lambda$. Obviously, the ES exhibits linear time complexity on the noisy sphere model, provided that the inequality in the evolution criterion is fulfilled.

Even though the performance analysis of the ES on the sphere yields valuable insight into the dynamical behavior of such strategies, there is still a need for investigations on more complex test functions. Especially, when the adaptation of the mutation operators is considered, the sphere model does not cover all essential aspects of the local evolution process. There is another class of simple fitness models that has been investigated empirically by Herdy [51]: the "ridge functions" with the special cases parabolic and sharp ridge (see also Rechenberg [52]). The general ridge function is defined by

$$f_{\mathrm{gr}}(\mathbf{x}) := \mathbf{v}^{\mathrm{T}}\mathbf{x} - d\left[\sqrt{[(\mathbf{v}^{\mathrm{T}}\mathbf{x})\mathbf{v} - \mathbf{x}]^2}\right]^{\alpha} \quad \text{with} \quad \mathbf{v}^{\mathrm{T}}\mathbf{v} = 1, \quad \alpha \ge 1,$$

where $\alpha = 1$ represents the sharp ridge and $\alpha = 2$ the parabolic ridge. The general ridge can be turned into its normal form by an orthogonal transformation rotating

$\mathbf{v}$, the so-called ridge direction, into a coordinate direction, say y_1. Thus, one obtains the normal form

$$f_\mathrm{r}(\mathbf{y}) := y_1 - dr^\alpha \quad \text{with} \quad r = \sqrt{\textstyle\sum_{i=2}^n y_i^2}.$$

Now it becomes clear that y_1 measures the projection of $\mathbf{x}$ on the ridge axis and r is the distance of $\mathbf{x}$ from the ridge axis. W.l.o.g. we assume a maximizing ES. Since r can only be reduced to zero but y_1 can grow infinitely, it is the general goal to evolve the population as fast as possible in (positive) y_1-direction. Starting from an arbitrary point in the $\mathbb{R}^n$, the amelioration process can be thought to be divided into two subgoals (Oyman [53]) minimizing r and enlarging y_1. In ES with isotropic mutations both subgoals are somewhat conflicting. As a result the analysis reveals a performance limit for $\alpha \geq 2$, although the success domain is an unbounded subset of $\mathbb{R}^n$.

Unlike the sphere model, where only one state variable was needed for describing the state of the $(\mu/\mu_I, \lambda)$-ES in the $\mathbb{R}^n$ search space, we now have to consider two state variables. As suggested by the normal form, y_1 and r are the appropriate variables to describe the evolution in $\mathbb{R}^n$. Therefore, the progress from one generation to the next must be evaluated by the two corresponding progress measures φ_y and φ_r, defined by

$$\varphi_y := \mathrm{E}(y_1^{(t+1)} - y_1^{(t)} \mid y_1^{(t)}, r^{(t)}, \sigma^{(t)})$$

and

$$\varphi_r := \mathrm{E}(r^{(t)} - r^{(t+1)} \mid y_1^{(t)}, r^{(t)}, \sigma^{(t)})$$

assuming isotropic (Gaussian) mutations with strength σ.

Deriving asymptotically exact progress rate expressions ($n \to \infty$) follows basically the ideas outlined for the sphere model. Since these formula are rather lengthy we don't want to rewrite them here (Oyman and Beyer [54], Beyer [55]). Instead, only the steady-state behavior will be discussed assuming an ES running with a fixed mutation strength σ. (Including the σ-adaptation in the analysis remains still to be done.)

As has already been mentioned, the amelioration process has to serve two conflicting subgoals. Depending on d, the amelioration of one of the subgoals can be emphasized. For example, when d is very large, one has basically an $(n-1)$-dimensional sphere model. The r-evolution is therefore governed by the sphere yielding a steady-state r_ss of r where

$$r_\mathrm{ss} \geq \frac{(n-1)\sigma}{2\mu c_{\mu/\mu,\lambda}};$$

note, the actually observed r_ss depends also on d and α.

For the steady-state progress rate in y_1-direction one finds

$$\varphi_{y_\mathrm{ss}} \simeq \frac{\sigma c_{\mu/\mu,\lambda}}{\sqrt{1 + (d\alpha r_\mathrm{ss}^{\alpha-1})^2}}.$$

As one can see, for $\sigma = \text{const.} > 0$, φ_{y_ss} cannot be negative, there is always a certain progress in y_1-direction. The sharp ridge case ($\alpha = 1$) yields a constant progress rate independent of r_ss. That is, increasing σ yields a linear increase in φ_{y_ss}. Recombination does not help in this case ($c_{\mu/\mu,\lambda} \leq c_{1,\lambda}$).

Things become more complicated when considering cases where $\alpha > 1$. The case $\alpha = 2$ is discussed by Oyman and Beyer [54]: one finds a steady-state φ_y that increases monotonously with σ approaching a performance maximum of $\mu c^2_{\mu/\mu,\lambda}/(n-1)d$ for $\sigma \to \infty$. Obviously, recombination increases the (generational) progress rate.

For the cases where $\alpha \in (0, 1)$, $\varphi_{y_{ss}}$ exhibits an unbounded increase with increasing σ, whereas, for $\alpha > 2$, $\varphi_{y_{ss}}$ runs through a maximum.

Since the r-dynamics reaches r_{ss} as steady-state value, the y_1-dynamics is mainly of interest here. Writing $Y^{(t)}$ for the expected value of $y_1^{(t)}$ one obtains $Y^{(t+1)} = Y^{(t)} + \varphi_{y_{ss}}(\sigma)$. Assuming constant σ, $\varphi_{y_{ss}}(\sigma)$ is constant, too. Provided that the ES has reached the steady-state regime after an initial time t_0, one gets

$$Y^{(t)} = Y^{(t_0)} + (t - t_0)\varphi_{y_{ss}}(\sigma).$$

That is, the ES travels linearly with the generation time along the ridge axis.

Calculating an estimate for the generations needed to travel a certain distance along the ridge axis is a trivial task. However, finding the n-dependency for $\alpha > 1$ needs further considerations. To this end, the normalizations

$$\varphi^\star := d^{\frac{1}{\alpha-1}}(n-1)\varphi_{y_{ss}}(\sigma) \quad \text{and} \quad \sigma^\star := d^{\frac{1}{\alpha-1}}(n-1)\sigma$$

are introduced. Using these normalizations and writing $T = t - t_0$, the difference equation for $Y^{(t)}$ can be solved for T :

$$T(n) = (n-1)\, d^{\frac{1}{\alpha-1}}\, \frac{Y^{(T+t_0)} - Y^{(t_0)}}{\varphi^\star(\sigma^\star)}.$$

For the $(1, \lambda)$-ES it has been shown asymptotically (Beyer [55]) that $\varphi^\star(\sigma^\star)$ does not depend on n and d. Therefore, one obtains a linear-time complexity. A similar behavior is expected for the general $(\mu/\mu_I, \lambda)$-ES, however, only the case $\alpha = 2$ has been investigated up to now.

3 New methods for discrete search spaces

In this section, we consider the optimization of (fitness) functions $f\colon S \to \mathbb{R}$ where the search spaces are finite. This is the domain of combinatorial optimization. In history, evolution strategies have been designed to work on infinite search spaces while genetic algorithms were designed for the optimization of pseudo-boolean functions $f\colon \{0,1\}^n \to \mathbb{R}$. Today, all variants of evolutionary algorithms are also applied for the optimization on discrete search spaces.

Since the very beginning researchers have contributed to a theory of evolutionary algorithms, although a great majority of all papers on evolutionary algorithms describes experimental results and develops rules of thumb. This experimental knowledge is immense and has a great influence on the actual application of evolutionary algorithms. However, the people working in theoretical computer science on efficient algorithms have (at least until recently) not worked on evolutionary algorithms and they have not accepted the theoretical results on evolutionary algorithms as "theory". Hence, in theoretical computer science evolutionary algorithms have been considered as the black sheeps in the family of algorithms. One knows that they exist and that they are applied (more or less successfully), but one has ignored them while developing a theory of efficient algorithms.

We try to explain the reasons for this situation. The scope of theoretical analysis of algorithms was for some time almost limited to deterministic algorithms, but since a long time randomized algorithms play a major rule (see, e.g., Motwani and Raghavan [56]). Also the aim of exact optimization has been supplemented by the aim of approximate optimization. Finally, theoretical computer science was focussed for a long time on the asymptotic behaviour. Several asymptotically very good algorithms have never been implemented, since they are too difficult to implement or since it was clear that they behave badly on instances of realistic size.

Nowadays, theory contributes to the area of algorithm engineering, i.e., to the design of algorithms which are easy to implement, efficient for instances of reasonable size, and asymptotically efficient. However, theory still insists on the analysis of algorithms. Algorithms should have a stamp of quality, i.e., the expected time to obtain a solution of a prescribed quality should be estimated as accurate as possible. Since it is most often impossible to obtain such results for each single instance of a problem, one considers the worst-case time with respect to classes of inputs which share some properties like the input length.

The classical contributions to a theory of evolutionary algorithms do not allow results of this type. There are many very precise results about what happens within one time period (generation) of an evolutionary algorithm. Performance measures like progress rate or quality gain are of this kind. Also the famous schema theorem is a statement about the one-step behavior. Moreover, there are many convergence results describing what happens as time goes to infinity. Some other results are obtained under unrealistic assumptions like the model of evolutionary algorithms working with populations of infinite size. Only Rabani, Rabinovich, and Sinclair [57] estimate the effect of such an assumption rigorously. However, their paper investigates a stochastic process without fitness-based selection. Finally, many attempts have been made to explain the working principle of GA as a building-block assembling strategy (Goldberg [27]): The final solution is obtained by successively putting together partial solutions through the application of the crossover operators. Given this picture, one can ask for bounds on the population size λ in order to guarantee for a correct assembly of the partial solutions (building blocks, assumed to be already existing in the initial population) with a certain error probability. This approach has been proposed by Goldberg, Deb, and Clark [58]. A population sizing theory based on a more refined model can be found in Harik, Cantú-Paz, Miller, and Goldberg [59].

Modeling binary GA as a dynamical system on a *macroscopic* level, i.e. by expected value dynamics (similar to the approach used in real-valued ES theory), has been proposed by Prügel-Bennett and Shapiro [60], Shapiro, Prügel-Bennett, and Rattray [61]. One of the basic ideas is to describe the population's fitness distribution by expansions of a Gaussian (also used in ES theory, see Beyer [62, 63]). The peculiarity of this approach is, however, that the underlying *microscopic* description level is bypassed using inference methods gleaned from statistical mechanics, especially the *maximum entropy principle* (Jaynes [64]). For an introduction into this interesting method as well as further references, the reader is referred to Prügel-Bennett and Rogers [65] and Shapiro [66].

Reviewing the history one may conclude that the theory on evolutionary algorithms has tried to obtain too general statements or too precise statements such that the results are limited to short periods of time or to the limit behavior. In particular, there were almost no results before the mid-nineties of the last century estimating the worst-case expected optimization time of an evolutionary algorithm working on some "problem" or estimating the probability of obtaining within $t(n)$ steps, $t(n)$ some polynomial, a solution fulfilling a minimum demand of quality.

During the last years attempts have been made to obtain such results also for evolutionary algorithms and to turn the theory on evolutionary algorithms into a legal part of the theory of efficient algorithms. This theory is still in its infancy. Here we describe only methods and results of this new approach. One has to admit that the analysis of evolutionary algorithms is somehow more difficult than the analysis of problem-specific algorithms. One reason is that many problem-specific algorithms have been designed not only to be efficient but also to allow a proof that they are efficient. Evolutionary algorithms have been designed to be successful in many situations and we have to analyze these fundamental variants of evolutionary algorithms.

Discrete optimization problems P consist of (typically infinitely many) "instances" or functions, each defined on a finite search space. The set of instances of the problem is partitioned into subclasses P_n which share the search space S_n. This includes all the famous combinatorial optimization problems. In this paper, we investigate pseudo-boolean functions where $S_n = \{0,1\}^n$. Our focus is not on classical combinatorial optimization problems like maximum matchings, maximum flow, shortest paths, or one of the many NP-equivalent problems. Instead of this we investigate classes of functions sharing some structural properties. This is motivated by the claim that evolutionary algorithms work efficiently on many types of problems as long as the resulting fitness functions have some "nice structure". In the future, one should also try to obtain results for classical optimization problems. Even for NP-equivalent optimization problems such an analysis is interesting, since one may consider simpler subproblems restricting the set of instances or restricting the demanded quality of the solution.

The rest of this section is organized as follows. First, we introduce the very simple $(1+1)$EA which is a mutation-based hill-climber working with population size 1 (this algorithm is also denoted as $(1+1)$ES). This algorithm is for many problems as efficient as all other mutation-based evolutionary algorithms and for its analysis we have to present many methods only used recently for the analysis of evolutionary algorithms.

Then we analyze the $(1+1)$EA on the class of linear functions, on the class of monotone pseudo-boolean functions of bounded degree, and on the class of unimodal functions. Then it is discussed what can be gained by more sophisticated variants of evolutionary algorithms which allow varying mutation probabilities, the use of multi-starts, or larger populations. Afterwards, we include the crossover operator and discuss problems to analyze evolutionary algorithms with crossover. We also describe how the 35 years old conjecture that crossover can decrease the expected optimization time from exponential to polynomial has been proved. For many of the considered problems it is obvious that problem-specific algorithms outperform evolutionary algorithms. However, the comparison of problem-specific algorithms and "general" evolutionary algorithms is unfair. The scenario of black-box optimization is described and it is argued that lower bounds on the black-box complexity of problems are lower bounds for "general" randomized search heuristics.

The $(1+1)$EA with mutation probability $1/n$ for the maximization of functions $f : \{0,1\}^n \to \mathbb{R}$.

1.) Choose $x \in \{0,1\}^n$ randomly with respect to the uniform distribution.

2.) Repeat until a stopping criterion is fulfilled:

   a) Construct the mutant x' from x where the bits $x'_1, \ldots, x'_n$ are created independently and $\text{Prob}(x'_i = x_i) = 1 - 1/n$.

   b) Replace x by x' iff $f(x') \geq f(x)$.

This algorithm is a randomized hill-climber, since x never is replaced by some x' with a worse fitness. Nevertheless, it cannot get stuck forever in a local optimum, since each $b \in \{0,1\}^n$ has a positive probability of at least n^{-n} to be the mutant of $a \in \{0,1\}^n$. This implies an upper bound of n^n for the expected optimization time of the $(1+1)$EA on any $f\colon \{0,1\}^n \to \mathbb{R}$. The $(1+1)$EA considered as an evolutionary algorithm is based on mutation and selection only.

We do not fix a stopping criterion, since we investigate the (1+1)EA without stopping criterion. For each f the random variable X_f describes the first point of time where some good event happens. In this paper the good event is the event that the current search point x is f-optimal. We are interested in the expected optimization time $E(X_f)$ and the success probability $\text{Prob}(X_f \le t)$. (This is another type of success probability as described in the Introduction.) For a problem P whose instances are described by the union of some $F_n \subseteq \{f\colon \{0,1\}^n \to \mathbb{R}\}$, the worst-case expected optimization time equals $t_P(n) := \max\{E(X_f) \mid f \in F_n\}$ and the worst-case success probability equals $s_{P,n}(t) := \min\{\text{Prob}(X_f \le t) \mid f \in F_n\}$.

Each pseudo-boolean function $f : \{0,1\}^n \to \mathbb{R}$ can be uniquely described as a polynomial

$$f(x) = \sum_{A \subseteq \{1,\ldots,n\}} w_A \cdot \prod_{i \in A} x_i.$$

Hence, we obtain the class $P(n,d,N)$ of all polynomials on n variables whose degree (the largest $|A|$ where $w_A \neq 0$) is bounded by d and where at most N terms have a non-zero weight (d and N may depend on n). A polynomial is called monotone increasing if no weight is negative. This description is not symmetric with respect to 0 or 1. We may describe f also with respect to $z = (z_1, \ldots, z_n)$ where $z_i = x_i$ for some i and $z_i = 1 - x_i$ for all other i. This new description of f has the same degree, but in general a different number of non-vanishing terms. A polynomial is called monotone if it is monotone increasing with respect to some $x \to z$-transformation.

Linear (or degree-1) polynomials can be maximized easily. The $(1+1)$EA does not explore the structure of the instance and depends on the instance f only via the f-values of all sample points x and x'. A good algorithm should be efficient on linear functions even if it does not "know" that the function is linear. Indeed, the linear function ONEMAX defined by $x_1 + \cdots + x_n$ was one of the first functions where the behavior of evolutionary algorithms has been analyzed. Mühlenbein [67] has proved that the expected optimization time is $O(n \log n)$. This analysis is based on a simple Markov chain approach. Droste, Jansen, and Wegener [68] have completed the analysis by a matching lower bound which is based on the coupon collector's theorem.

It is already rather complicated to analyze the (1+1)EA on all linear functions. Droste, Jansen, and Wegener [69] have proved the following result.

The expected optimization time of the $(1+1)$EA on a linear function is bounded above by $O(n \log n)$ and, if all weights are non-zero, bounded below by $\Omega(n \log n)$.

The lower bound follows easily from the coupon collector's theorem. For the upper bound it is also easy to see that it is sufficient to consider monotone increasing linear functions. It is illustrative to discuss first the proof idea for the function binary value defined by

$$BV(x) = x_1 \cdot 2^{n-1} + x_2 \cdot 2^{n-2} + \cdots + x_n \cdot 2^0.$$

The leftmost flipping bit decides whether the mutant is better than its parent. The Hamming distance to the optimum 1^n can be increased, the mutant 10^{n-1} of 01^{n-1} is accepted. However, the number of flipping bits has an average number of 1 and we expect that successful steps (x' replaces x) tend to increase the number of ones. We partition the run of the $(1+1)$EA into phases where the i-th phase starts with the first search point with at least i ones. In a successful step, the leftmost flipping bit is a 0-bit. We assume pessimistically that all bits to the right of the leftmost flipping bit are ones. If these are d bits, we may increase the number of ones at most by 1, but may decrease it significantly. Again pessimistically we assume that d takes its largest possible value $n-1$. Then we investigate a homogeneous random walk on a line starting at i. We are interested in the first point of time where we reach $i+1$ (by our assumption steps to the right are steps of length 1 and we cannot jump over $i+1$). If the expected step length is bounded below by a positive constant c, we obtain by Wald's identity the result that the expected waiting time is bounded above by $1/c$, also a constant. However, in our situation the expected step length of the random walk equals $1/n$. For the special case of the function BV we can ignore at first the right half of the string, since these bits do not influence the $(1+1)$EA on the left half of the string. Then the number of bits flipping from 1 to 0 in a successful step is bounded above by $n/2-1$ leading to an expected step length of at least $1/2$ and an expected number of at most two successful steps within one phase. The probability of a successful step in the i-th phase is at least $(n/2-i)/(en)$, since there are at least $n/2-i$ zeros in the left half and $n/2-i$ 1-bit mutations lead to successful steps. The probability of each specific 1-bit mutation in the left half equals $\frac{1}{n}(1-\frac{1}{n})^{n/2-1} \geq 1/(en)$. This leads to a bound of $O(n \log n)$ until the left half consists of ones only. A similar analysis works for the right half where only steps without flipping bit in the left half are successful.

This description of the proof idea shows that we have to be satisfied with a O-bound, since only pessimistic assumptions lead to a Markov chain which can be analyzed. However, the resulting constant factor is still not much larger than the constant factor in the corresponding lower bound.

We have described the proof for the specific example BV as an illustrative example. A general linear function may have some weights which are almost equal (as ONEMAX) but also extremely different weights (as BV). For the analysis we can assume that $w_1 \geq \cdots \geq w_n$. However, it is not possible to ignore the right half of the bits in the first phase. Droste, Jansen, and Wegener [69] have used the idea to measure the progress of the optimization procedure by a potential function which in this case itself is linear namely $g(x) := 2(x_1 + \cdots + x_{n/2}) + (x_{n/2+1} + \cdots + x_n)$. The $(1+1)$EA works on an arbitrary linear function f but the progress is measured with respect to the potential function g. This implies that the g-value can decrease during the optimization process. Again the idea is to prove a bound of $O(1)$ on the expected number of steps until from the g-value of i a g-value of at least $i+1$ is reached. It is then easy to finish the proof by an investigation of the expected number of unsuccessful steps. Here it is already technically involved to define a Markov chain which provably is slower than the $(1+1)$EA with respect to the potential function and which nevertheless allows the proof that the expected gain of the g-value is bounded below by a positive constant.

The discussion shows that even the asymptotically exact analysis of evolutionary algorithms on simple classes like the class of linear functions is technically involved. Similarly to the analysis of many problem-specific algorithms one has to obtain an intuition how the optimization works and then apply the tool-box for the analysis of algorithms. Already the analysis of the $(1+1)$EA on linear functions is quite different from the typical papers on the theory of evolutionary algorithms and it applies ideas quite similar to those used in the analysis of problem-specific randomized algorithms.

The next step is to investigate polynomials of higher degree. However, it is well known that the maximization of pseudo-boolean polynomials of degree 2 is NP-hard. Hence, we do not expect that any randomized search heuristic has a polynomially bounded expected optimization time. Whenever an algorithm for a problem has an exponential worst-case behavior, one is interested in particular instances where this algorithm fails. Droste, Jansen, and Wegener [70] have defined a family of degree-3 polynomials based on instances for MAX-3-SAT where the $(1+1)$EA and all generic variants of evolutionary algorithms need exponential time on the average. They even have proved that the success probability of each mutation-based evolutionary algorithm producing $\exp(o(n^{1/2}))$ strings is bounded by $\exp(-\Omega(n^{1/2}))$. This instance (see also Papadimitriou [71]) consists of the clauses $x_i, 1 \leq i \leq n$, and $x_i + \overline{x}_j + \overline{x}_k, 1 \leq i, j, k \leq n, i \neq j \neq k \neq i$. The string 1^n is the only one to fulfil all clauses, but as long as the number of 1-bits is larger than 1 and less than $(2/3)n$ one increases the number of satisfied clauses by flipping a 1-bit and not by flipping a 0-bit.

Wegener and Witt [72] have defined a degree-2 polynomial where the $(1+1)$EA has the following behavior:

- for each $\varepsilon > 0$, the success probability within $c(\varepsilon)n \log n$ steps, $c(\varepsilon)$ a constant, is at least $1/2 - \varepsilon$,
- for each $\varepsilon > 0$, the success probability within $2^{o(n \log n)}$ steps is bounded above by $1/2 + \varepsilon$.

Hence, the expected optimization time is exponential. The function is defined by

$$f(x_1, \ldots, x_n, y) := y \cdot 2 \cdot (x_1 + \cdots + x_n) + (1 - y) \cdot ((3n/2) - x_1 - \cdots - x_n).$$

The value of y decides whether we like to maximize the number of 0-bits or the number of 1-bits of x. The first value of y is random and it is quite likely that it changes only after several steps. Then the x-string has with large probability many more ones then zeros (if $y = 1$) or vice versa (if $y = 0$). Then many x-bits have to flip in one step to accept a change of y. Hence, starting with $y = 1$ it is likely to reach the optimal string 1^n quickly, but starting with $y = 0$, it is likely to reach the sub-optimal string 0^n and then we have to wait for a long time until we leave this local optimum. It is obvious that in such a situation a multi-start variant of the $(1+1)$EA works efficiently even in the expected case. Using $p(n) = \omega(n \log n)$ independent runs of the (1+1)EA, the expected optimization time is bounded by $O(p(n) \cdot n \cdot \log n)$. Here we assume that during one phase each run of the $(1+1)$EA performs one step and that the cost of a phase is $p(n)$. One may expect that also evolutionary algorithms with a larger population size than 1 work efficiently. This is still open, since in that situation the strings of the current population do not act independently. The probability that selection pressure forces all strings to the neighborhood of 0^n can be not negligible.

These examples reveal certain difficulties of evolutionary algorithms when optimizing low-degree polynomials with a simple description. This makes it interesting to consider classes of polynomials where evolutionary algorithms are efficient.

A simple case is the optimization of positive monomials $w \cdot z_{i(1)} z_{i(2)} \cdots z_{i(d)}$ where $w > 0, 1 \le i(1) < \cdots < i(d) \le n$ and $z_i \in \{x_i, 1 - x_i\}$. The expected optimization time of the $(1+1)$EA on such monomials is $\Theta(2^d n/d)$. This can be proved by a direct Markov chain approach, since the $(1+1)$EA accepts each string until it finds an optimal string. Garnier, Kallel, and Schoenauer [73] have performed the analysis for $d = n$, but their results can be generalized easily. The result is not surprising. W.l.o.g. we discuss the case of $x_1 x_2 \cdots x_d$. There are 2^d assignments to $(x_1, \ldots, x_d)$, $2^d - 1$ of them cannot be distinguished by the value of the monomial and the last one is optimal. We cannot expect to find the optimal assignment (without knowing the monomial) in $o(2^d)$ steps. The additional factor describes the average number of steps until the $(1+1)$EA flips one of the d essential bits.

Now we investigate monotone polynomials which are sums of positive monomials where $z_i = x_i$ in all monomials or $z_i = 1 - x_i$ in all monomials. The following result can be proved.

The expected optimization time of the $(1+1)$EA on monotone polynomials with N non-vanishing terms and degree $d \le \log n$ is bounded by $O(Nn2^d/d)$.

The special case of $d = 2$ has been considered by Wegener and Witt [72] and the more general case by Wegener [74]. We describe some ideas of the proof. We assume that the terms are numbered in such a way that $w_1 \ge \cdots \ge w_N > 0$ for the weight w_i of term t_i. Moreover, we assume that the terms contain only positive variables x_j. The evolutionary algorithm has to activate all terms. An assignment a activates a term t iff $t(a) > 0$. On one hand it is useful to have many activated terms, on the other hand it is useful that the important terms (with large weight) are activated. It may happen that the number of activated terms decreases and it also may happen that the most important term is deactivated.

In order to cope with these problems we partition $\{0,1\}^n$ into $N+1$ fitness layers $L_0, \ldots, L_N$ where

$$L_i := \{a | w_1 + \cdots + w_i \leq f(a) < w_1 + \cdots + w_{i+1}\}$$

and $L_N := \{a | f(a) = w_1 + \cdots + w_N\}$ contains all optimal strings. In order to prove the proposed bound it is sufficient to prove that the expected time to leave L_i is bounded by $O(n2^d/d)$ which is the time bound to activate a single monomial. Indeed, for $a \in L_i$ there exists a passive term whose activation would imply that we leave L_i. Therefore, we consider the period of time until this special term is activated.

We have to cope with the influence of the other terms:

- There are steps which are not accepted by the (1+1)EA, since the total fitness decreases.

- The step activating the special term may be not accepted, since other terms are deactivated.

It is straightforward to show that the last event happens with a probability bounded by a constant less than 1. If this happens, we start another trial (with perhaps another special term). However, we need only an expected number of $O(1)$ trials. The first event can happen in many steps. Intuitively, we believe that monotonicity is essential in this situation. The fitness decreases only if a term is deactivated. This implies that a 1-bit flips and this is a bad event for the optimization of the polynomial. In particular, if a 1-bit flips which is contained in our special term, we are happy if such a step is not accepted. These arguments may sound convincing, but they are far from a proof.

The proof is done by comparing the activation of the special term with the $(1+1)$EA working on the given polynomial with the activation of the special term with the (1+1)EA working on this term only. The idea is to prove a lower bound of some positive constant on the probability that the term is activated within $c \cdot 2^d \cdot n/d$ steps (c some appropriate constant). Then the expected number of these phases is $O(1)$. It is rather unlikely that three bits of the term flip simultaneously within one phase and we ignore phases with such steps. Then we are in the situation of quite local changes which can be handled by a tedious analysis.

This analysis is based on super-pessimistic assumptions. We wait for a special term to be activated in order to leave a fitness layer. We leave this layer also if other terms or small groups of terms are activated. Moreover, monotonicity leads to a positive correlation for the activation of terms which share variables. Hence, we conjecture that among the monotone polynomials those with disjoint terms are the most difficult ones. Let $n = dk$. Then

$$x_1 x_2 \cdots x_d + x_{d+1} x_{d+2} \cdots x_{2d} + \cdots + x_{n-d+1} x_{n-d+2} \cdots x_n$$

has been called royal road function by Mitchell, Forrest, and Holland [75]. They have assumed, based on the building-block hypothesis, that crossover is essential for the optimization of this function. However, they have proved later (Mitchell, Holland, and Forrest [76]) that the $(1+1)$EA is very efficient on this function. Its expected optimization time equals $\Theta(2^d \cdot (n/d) \log(n/d))$. We conjecture that the bound $O(2^d \cdot n \cdot \log n)$ holds for all monotone polynomials of degree d. It is a challenge to prove this bound.

Royal road functions are one example where even the intuition of those who have invented genetic algorithms failed. Only the analysis of the expected optimization time can decide which algorithm is the better one in a given situation.

A pseudo-boolean function is called unimodal if the global optimum is unique and all other points of the search space have a better Hamming neighbor. The discussion whether unimodal functions can be optimized efficiently by evolutionary algorithms has a long history. Horn, Goldberg, and Deb [77] have introduced unimodal long path functions with the following properties. It is easy to find the starting point of a path $p_0, p_1, \ldots, p_l$ where $H(p_i, p_{i+1}) = 1$ (the Hamming distance), $f(p_{i+1}) > f(p_i)$, and for all points x outside the path $f(x) < f(p_0)$. They also presented examples where experiments led to the conjecture that the expected optimization time is exponential. Also numerical investigations by Höhn and Reeves [78] could not solve the problem. Finally, Rudolph [79] proved that the $(1+1)$EA typically does not follow the path and uses shortcuts which leads to an expected optimization time of $O(n^3)$. He also designed another long path function where shortcuts seem to be unlikely. Droste, Jansen, and Wegener [80] have proved this conjecture by proving that all mutation-based evolutionary algorithms need on the average exponential time on this problem. Some experiments show that crossover may be helpful for some long path problems. Others may add more and more search operators and the discussion can be stopped only by proving that no randomized search heuristic is efficient on all unimodal functions.

For such a purpose, we have to consider a class of algorithms including all general randomized search heuristics. This is the black-box scenario. A general heuristic does not work with the parameters of the function which are assumed to be unknown. It can obtain information about f only via sampling. In the t-th step the algorithm knows the first $t-1$ sample points $a_1, \ldots, a_{t-1} \in \{0,1\}^n$ and their fitness values $f(a_1), \ldots, f(a_{t-1})$ and it may decide about a probability distribution to choose $a_t \in \{0,1\}^n$.

A black-box algorithm is very powerful, since all calculations are free, only sampling is charged. However, it is limited, since the parameters of the instance are not known. In order to prove lower bounds for randomized algorithms, one can apply Yao's minimax principle [81]. If the number of instances is finite, a lower bound for the average-case optimization time of deterministic algorithms with respect to a given probability distribution on the inputs is a lower bound for the worst-case expected optimization time for all randomized algorithms. Droste, Jansen, Tinnefeld, and Wegener [68] have defined a probability distribution on a finite subset of all unimodal functions on $\{0,1\}^n$ such that the average-case optimization time of deterministic algorithms is exponential. Since general evolutionary algorithms and all other general randomized search heuristics are black-box algorithms, we know that none of them is efficient on all unimodal functions. This claim holds independently from complexity-theoretical assumptions like NP$\neq$RP.

The (1+1)EA is a very simple evolutionary algorithm. We have already seen that multi-start variants can outperform the (1+1)EA significantly. Jansen and Wegener [82] have also proved that one may need not too small populations in order to obtain good expected optimization times – even in the absence of the crossover operator.

The (1+1)EA has been described with a mutation probability of $1/n$, but some lower bounds were stated for all mutation-based evolutionary algorithms. This allows mutation probabilities different from $1/n$ and even varying over time as long as in each mutation step each bit is flipped with the same probability. Sometimes it has been conjectured that no mutation probability is significantly better than $1/n$. This has been proved for certain functions (Bäck [83], Mühlenbein [67]). In applications, several heuristics to choose good mutation probabilities adaptively have been suggested.

Our analytical tools can clarify these questions (Droste, Jansen, and Wegener [84], Jansen and Wegener [85, 86]). There are functions where only mutation probabilities of the unusual size $\Theta((\log n)/n)$ lead to a polynomial expected optimization time. In other situations each fixed mutation probability leads to an exponential optimization time while a simple dynamic schedule guarantees a polynomial expected optimization time. This schedule starts with a mutation probability of $1/n$, doubles it after each step until a value of at least $1/2$ is reached. Then the parameter is reset to $1/n$ and a new phase starts. Although steps with a large mutation probability may cause a disaster, this dynamic variant is quite robust, for many functions the expected optimization time does not increase significantly.

Population size 1 with or without the multi-start option simplifies the analysis. Populations lead to individuals depending on each other. The so-called hitchhiking effect is based on the observation that often the individuals of a population share many bits. Then it is wasteful to work with a population instead of a single individual. Crossover has only a chance to contribute essentially to the search if it is applied to quite different strings. Very early attempts to consider the effect of crossover empirically are described by Rechenberg [4].

Jansen and Wegener [87] have investigated a population on the k-th level of $\{0,1\}^n$, i.e., only strings with exactly k ones are accepted. Mutation has the effect to create new strings. However, the mutants do not differ much from their parents. Crossover of x and y creates a string z "between x and y". If k is far away from 0 and n, the level is much larger than polynomial and one may hope that crossover and mutation create a population with quite different strings. Such a result has been proved in Jansen and Wegener [87] for quite small crossover probabilities. This paper reveals the problems of such an analysis. The results have been obtained under the very pessimistic assumption that crossover always has the worst possible result.

The aim of Jansen and Wegener [87] was not a general analysis of the crossover operator. The aim was much more pragmatic. Crossover has been used since the sixties of the last century and its usefulness has been discussed in numerous papers. Despite these investigations no function was known where crossover reduces the expected optimization time from non-polynomial to polynomial. The analysis of Jansen and Wegener [87] leads to the first result of this type. However, the expected optimization time is reduced only from superpolynomial to polynomial and this can be proved only for crossover probabilities of size $o(1/n)$. The results have been improved in Jansen and Wegener [88] for other artificial functions. These are functions called real royal road functions (different functions depending on the crossover type where the most popular types namely one-point crossover and uniform crossover are considered) such that all mutation-based evolutionary algorithms have an exponential expected optimization time while a genetic algorithm only needs polynomial time on the average. The results hold for all crossover probabilities not too close to 0 and 1 (at least $1/p(n)$ and at most $1 - 1/p(n)$ for some polynomial p), for all reasonable selection procedures to select parents (the probability of choosing x as parent is at least as large as the corresponding probability for y if $f(x) \geq f(y)$) and a reasonable procedure to select the next generation (choose the string with the smallest fitness and, if there are many strings with the same fitness, avoid duplicates as much as possible). These results are not very difficult to prove after one has chosen the appropriate functions and has the tool-box described in this section.

We conclude that (at least for discrete search spaces) one can apply for the analysis of evolutionary algorithms methods developed for the analysis of problem-specific randomized algorithms. This leads to upper and lower bounds on the expected optimization time and the success probability of evolutionary algorithms. Such results are necessary to compare evolutionary algorithms with their competitors. The results should not hide the fact that the analysis of the expected optimization time of evolutionary algorithms is still in its infancy. Many more problems are open than solved. It is our hope that the approach described in this section motivates more researchers to work in this direction.

References

[1] H.-P. Schwefel. Direct search for optimal parameters within simulation models. In R. D. Conine, E. D. Katz, and J. E. Melde, editors, *Proc. Twelfth Annual Simulation Symp.*, pages 91–102. IEEE Computer Society, Long Beach CA, 1979.

[2] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution.* Wiley, New York, 1966.

[3] J. H. Holland. *Adaptation in natural and artificial systems.* The Univ. of Michigan Press, Ann Arbor, MI, 1975.

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

[5] H.-P. Schwefel. *Numerische Optimierung von Computer–Modellen mittels der Evolutionsstrategie*, volume 26 of *Interdisciplinary Systems Research.* Birkhäuser, Basle, Switzerland, 1977.

[6] editor D. B. Fogel. *Evolutionary Computation — The Fossil Record.* IEEE Press, Piscataway NJ, 1998.

[7] H.-P. Schwefel and R. Männer, editors. *Parallel Problem Solving from Nature – Proc. First Workshop PPSN*, volume 496 of *Lecture Notes in Computer Science.* Springer, Berlin, 1991.

[8] R. K. Belew and L. B. Booker, editors. *Proc. Fourth Int'l Conf. Genetic Algorithms (ICGA'91).* Morgan Kaufmann, San Mateo CA, 1991.

[9] D. B. Fogel and W. Atmar, editors. *Proc. First Annual Conf. Evolutionary Programming (EP'92).* Evolutionary Programming Society, San Diego CA, 1992.

[10] Th. Bäck, D. B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation.* Oxford University Press, New York, and Institute of Physics Publishing, Bristol, 1997.

[11] E. Fiesler and R. Beale, editors. *Handbook of Neural Computation.* Oxford University Press, New York, and Institute of Physics Publishing, Bristol, 1997.

[12] E. H. Ruspini, P. P. Bonissone, and W. Pedrycz, editors. *Handbook of Fuzzy Computation.* Institute of Physics Publishing, Bristol, 1998.

[13] Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors. *Proc. First IEEE Conf. Evolutionary Computation*, First IEEE World Congress on Computational Intelligence. IEEE Press, Piscataway NJ, 1994.

[14] D. B. Fogel, H.-P. Schwefel, Th. Bäck, and X. Yao, editors. *Proc. Fifth IEEE Conf. Evolutionary Computation*, Second IEEE World Congress on Computational Intelligence. IEEE Press, Piscataway NJ, 1998.

[15] H. G. Beyer, E. Brucherseifer, W. Jakob, H. Pohlheim, B. Sendhoff, and T. B. To. Evolutionäre Algorithmen — Begriffe und Definitionen. Technical Report of the Collaborative Research Center 531 *Computational Intelligence* CI–115/01, University of Dortmund, 2001.

[16] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press, New York, 1995.

[17] D. B. Fogel. On the philosophical differences between evolutionary algorithms and genetic algorithms. In D. B. Fogel and W. Atmar, editors, *Proc. Second Annual Conf. Evolutionary Programming (EP'93)*, pages 23–29. Evolutionary Programming Society, San Diego CA, 1993.

[18] L. J. Eshelman and J. D. Schaffer. Real-coded genetic algorithms and interval schemata. In L. D. Whitley, editor, *Proc. of Foundations of Genetic Algorithms*, volume 2, pages 187–202, 1991.

[19] H. Mühlenbein and D. Schlierkamp-Voosen. The science of breeding and its application to the breeder genetic algorithm I: Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49, 1993.

[20] J. T. Alander. Bibliography of genetic algorithms. In J. T. Alander, editor, *Proc. Second Finnish Workshop on Genetic Algorithms and their Applications*, pages 1–22. University of Vaasa, Finland, 1994.

[21] H.-P. Schwefel. *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik.* Dipl.-Ing. Thesis, Technical University of Berlin, Hermann Föttinger–Institute for Hydrodynamics, March 1965.

[22] H.-P. Schwefel. *Numerical Optimization of Computer Models.* Wiley, Chichester, 1981.

[23] H.-P. Schwefel. Collective phenomena in evolutionary systems. In P. Checkland and I. Kiss, editors, *Problems of Constancy and Change – The Complementarity of Systems Approaches to Complexity, Papers presented at the 31st Annual Meeting Int'l Soc. General System Research*, volume 2, pages 1025–1033, Budapest, June 1–5, 1987. Int'l Soc. for General System Research.

[24] G. Rudolph. On correlated mutations in evolution strategies. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature, 2*, pages 105–114. North Holland, Amsterdam, 1992.

[25] G. Rudolph. *Convergence Properties of Evolutionary Algorithms.* Verlag Dr. Kovač, Hamburg, 1997.

[26] J. Maynard Smith. Models of evolution. *Proc. Royal Soc. London B*, 219:315–325, 1983.

[27] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison–Wesley, Reading MA, 1989.

[28] J. J. Grefenstette. Deception considered harmful. In L. D. Whitley, editor, *Foundations of Genetic Algorithms, 2*, pages 75–91, San Mateo CA, 1993. Morgan Kaufmann.

[29] H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. Natural Computing 1. In print.

[30] X. Qi and F. Palmieri. Adaptive mutation in the genetic algorithms. In *Proceedings of the 2nd Annual Conference on Evolutionary Programming*, pages 192–196, La Jolla (CA), 1993. Evolutionary Programming Society.

[31] H.-G. Beyer and K. Deb. On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 5(3):250–270, 2001.

[32] W. M. Spears. *Evolutionary Algorithms: The Role of Mutation and Recombination*. Springer, Heidelberg, 2000.

[33] M. D. Vose. *The Simple Genetic Algorithm: Foundations and Theory*. MIT Press, Cambridge, MA, 1999.

[34] J. Born. *Evolutionsstrategien zur numerischen Lösung von Adaptationsaufgaben*. Ph.D. Thesis, Humboldt-Universität, Berlin, 1978.

[35] K. De Jong. Are genetic algorithms function optimizers? In *Proc. of PPSN II (Parallel Problem Solving from Nature)*, pages 3–13, Amsterdam, 1992. North Holland.

[36] K. De Jong. Genetic algorithms are not function optimizers. In *Foundations of Genetic Algorithms, 2*, pages 5–17, San Mateo, CA, 1993. Morgan Kaufmann.

[37] G. Rudolph. Convergence properties of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1):96–101, 1994.

[38] T. E. Davis and J. C. Principe. A Markov chain framework for the simple genetic algorithm. *Evolutionary Computation*, 1(3):269–288, 1993.

[39] G. Rappl. *Konvergenzraten von Random Search Verfahren zur globalen Optimierung*. Ph. D. Thesis, HSBw München, Germany, 1984.

[40] G. Rappl. On linear convergence of a class of random search algorithms. *Zeitschrift f. angew. Math. Mech. (ZAMM)*, 69(1):37–45, 1989.

[41] D. V. Arnold and H.-G. Beyer. Noisy optimization with evolution strategies. SIAM Journal on Optimization. Submitted for publication.

[42] D. V. Arnold and H.-G. Beyer. Local performace of the $(\mu/\mu_I, \lambda)$-ES in a noisy environment. In *Proc. of Foundations of Genetic Algorithms, 6*, pages 127–141, San Francisco, CA, 2001. Morgan Kaufmann.

[43] H.-G. Beyer. *The Theory of Evolution Strategies*. Natural Computing Series. Springer, Heidelberg, 2001.

[44] H.-G. Beyer and D. V. Arnold. Fitness noise and localization errors of the optimum in general quadratic fitness models. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 817–824, San Francisco, CA, 1999. Morgan Kaufmann.

[45] H.-G. Beyer. An alternative explanation for the manner in which genetic algorithms operate. *BioSystems*, 41:1–15, 1997.

[46] D. V. Arnold and H.-G. Beyer. Performance analysis of evolution strategies with multi-recombination in high-dimensional $\mathbb{R}^N$-search spaces disturbed by noise. Theoretical Computer Science. Accepted for publication.

[47] H.-G. Beyer. Toward a theory of evolution strategies: Self-adaptation. *Evolutionary Computation*, 3(3):311–347, 1996.

[48] D. V. Arnold. *Local Performance of Evolution Strategies in the Presence of Noise*. Ph.D. Thesis, University of Dortmund, Dortmund, 2001.

[49] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proc. of IEEE Int'l Conf. on Evolutionary Computation (ICEC '96)*, pages 312–317. IEEE Press, NY, 1996.

[50] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[51] M. Herdy. Reproductive isolation as strategy parameter in hierarchically organized evolution strategies. In *PPSN II (Parallel Problem Solving from Nature)*, pages 207–217, Amsterdam, 1992. Elsevier.

[52] I. Rechenberg. *Evolutionsstrategie '94*. Frommann-Holzboog Verlag, Stuttgart, 1994.

[53] A. I. Oyman. *Convergence Behavior of Evolution Strategies on Ridge Functions.* Ph.D. Thesis, University of Dortmund, Dortmund, 1999.

[54] A. I. Oyman and H.-G. Beyer. Analysis of the $(\mu/\mu, \lambda)$-ES on the parabolic ridge. *Evolutionary Computation*, 8(3):267–289, 2000.

[55] H.-G. Beyer. On the performance of $(1, \lambda)$-evolution strategies for the ridge function class. *IEEE Transactions on Evolutionary Computation*, 5(3):218–235, 2001.

[56] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambrige University Press, 1995.

[57] Y. Rabani, Y. Rabinovich, and A. Sinclair. A computational view of population genetics. *Random Structures and Algorithms*, 12:314–334, 1998.

[58] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6(4):333–362, 1992.

[59] G. R. Harik, E. Cantú-Paz, B. L. Miller, and D. E. Goldberg. The gambler's ruin problem, genetic algorithms, and the sizing of populations. In *Proc. of 1997 IEEE Int'l Conf. on Evolutionary Computation (ICEC '97)*, pages 7–12, Indianapolis IN, 1997. IEEE Press, Piscataway NJ.

[60] A. Prügel-Bennett and J. L. Shapiro. An analysis of genetic algorithms using statistical mechanics. *Phys. Rev. Lett.*, 72(9):1305, 1994.

[61] J. L. Shapiro, A. Prügel-Bennett, and L.M. Rattray. A statistical mechanical formulation of the dynamics of genetic algorithms. In *Lecture Notes in Computer Science, Vol. 865*, pages 17–27. Springer, Heidelberg, 1994.

[62] H.-G. Beyer. Towards a theory of 'evolution strategies': Progress rates and quality gain for $(1\stackrel{+}{,}\lambda)$-strategies on (nearly) arbitrary fitness functions. In *PPSN II (Parallel Problem Solving from Nature)*, pages 58–67, Heidelberg, 1994. Springer.

[63] H.-G. Beyer. Toward a theory of evolution strategies: The (μ, λ)-theory. *Evolutionary Computation*, 2(4):381–407, 1995.

[64] E. T. Jaynes. Where do we stand on maximum entropy? In R.D. Levine and M. Tribus, editors, *The Maximum Entropy Formalism*, pages 15–118, Cambridge, MA, 1979. MIT Press.

[65] A. Prügel-Bennett and A. Rogers. Modelling genetic algorithm dynamics. In *Proc. of 2nd EvoNet Summer School on Theoretical Aspects of Evolutionary Computing*, pages 59–85. Springer, Heidelberg, 2001.

[66] J. L. Shapiro. Statistical mechanics theory of genetic algorithms. In *Proc. of 2nd EvoNet Summer School on Theoretical Aspects of Evolutionary Computing*, pages 87–108. Springer, Heidelberg, 2001.

[67] H. Mühlenbein. How genetic algorithms really work. I. mutation and hillclimbing. In *Proc. of PPSN II (Parallel Problem Solving from Nature)*, pages 15–25, 1992.

[68] S. Droste, T. Jansen, K. Tinnefeld, and I. Wegener. A new framework for the valuation of algorithms for black-box optimization. 2001. Preprint.

[69] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science. Accepted for publication.

[70] S. Droste, T. Jansen, and I. Wegener. A natural and simple function which is hard for all evolutionary algorithms. In *Proc. of IECON'2000 (IEEE Int. Conf. on Industrial Electronics, Control and Instrumentation)*, pages 2704–2709, 2000.

[71] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[72] I. Wegener and C. Witt. On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. Journal of Discrete Algorithms. Submitted for publication.

[73] J. Garnier, L. Kallel, and M. Schoenauer. Rigorous hitting times for binary mutations. *Evolutionary Computation*, 7:173–203, 1999.

[74] I. Wegener. Theoretical aspects of evolutionary algorithms. In *Proc. of ICALP'2001*, LNCS 2076, pages 64–78, 2001.

[75] M. Mitchell, S. Forrest, and J. H. Holland. The royal road function for genetic algorithms: Fitness landscapes and GA performance. In *Proc. of 1st European Conf. on Artificial Life*, pages 245–254. MIT, 1994.

[76] M. Mitchell, J. H. Holland, and S. Forrest. When will a genetic algorithm outperform hill climbing? Advances in Neural Information Processing Systems.

[77] J. Horn, D. E. Goldberg, and K. Deb. Long path problems. In *Proc. of PPSN III (Parallel Problem Solving from Nature)*, LNCS 866, pages 149–158, 1994.

[78] C. Höhn and C. Reeves. Are long path problems hard for genetic algorithms? In *Proc. of PPSN IV (Parallel Problem Solving from Nature)*, LNCS 1141, pages 134–143, 1996.

[79] G. Rudolph. How mutation and selection solve long-path problems in polynomial expected time. *Evolutionary Computation*, 4:64–78, 1997.

[80] S. Droste, T. Jansen, and I. Wegener. On the optimization of unimodal functions with the $(1+1)$ evolutionary algorithm. In *Proc. of PPSN V (Parallel Problem Solving from Nature)*, LNCS 1498, pages 13–22, 1998.

[81] A. C. Yao. Probabilistics computations: Towards a unified measure of complexity. In *Proc. of 17th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 222–227, 1977.

[82] T. Jansen and I. Wegener. On the utility of populations in evolutionary algorithms. In *Proc. of GECCO'2001 (Genetic and Evolutionary Computation Conference)*, pages 1034–1041, 2001.

[83] T. Bäck. Optimal mutation rates in genetic search. *Proc. of 5th Int. Conference on Genetic Algorithms*, pages 2–8, 1993.

[84] S. Droste, T. Jansen, and I. Wegener. Dynamic parameter control in simple evolutionary algorithms. In *Proc. of Foundations of Genetic Algorithms*, volume 6, pages 275–294, 2001.

[85] T. Jansen and I. Wegener. On the choice of the mutation probability for the (1+1)EA. In *Proc. of PPSN VI (Parallel Problem Solving from Nature)*, LNCS 1917, pages 89–98, 2000.

[86] T. Jansen and I. Wegener. On the analysis of a dynamic evolutionary algorithm. Journal of Discrete Algorithms. Submitted for publication.

[87] T. Jansen and I. Wegener. On the analysis of evolutionary algorithms – a proof that crossover really can help. In *Proc. of 7th European Symp. on Algorithms (ESA'99)*, LNCS 1643, pages 184–193, 1999.

[88] T. Jansen and I. Wegener. Real royal road functions – where crossover provably is essential. In *Proc. of GECCO'2001 (Genetic and Evolutionary Computation Conference)*, pages 375–382, 2001.