Mutation by Imitation in Boolean Evolution Strategies

Michèle Sebag[1] and Marc Schoenauer[2]

(1) LMS – URA CNRS 317 (2) CMAP – URA CNRS 756
Ecole Polytechnique, 91128 Palaiseau Cedex, France
`Michele.Sebag@polytechnique.fr` `Marc.Schoenauer@polytechnique.fr`

Abstract. Adaptive heuristics have been developed in the Evolution Strategy (ES) frame regarding the mutation of real-valued variables. But these heuristics poorly extend to discrete variables: when the rate or variance of mutation gets too small, mutation has no effect any more. To overcome this problem, we propose two mutation operators, that use the worst individuals of the current population as beacons indicating the limits of the current promising region:
Mutation by differentiation drives individuals away from the beacon-individuals. *Mutation by imitation* inversely assumes that beacon-individuals still contain relevant informations, and aims at moving the individual at hand nearer to the beacons. Mutation by imitation produces offspring that share the features of several "parents"; but in contrast with classical crossover, it allows one to control the distance between the offspring and the main parent, by fixing the number of bits to mutate. Mutation by imitation thus permits a tunable exchange of informations among individuals.
Both operators have been implemented in a boolean $(\mu + \lambda)$ ES framework, and experimented on four problems: the Royal Road problem, a GA-deceptive problem, the combinatorial multiple knapsack optimization problem and the Long Path problem. Comparative validation is presented and discussed.

1 Introduction

The three historical roots of evolutionary computation are known to be evolutionary programming (EP), evolution strategies (ES) and genetic algorithms (GA). Original ES were developed in the mid sixties [24, 23] to deal with real parameter optimization. EP was initially developed in the context of Finite State Machines [7], but was later modified and tuned to handle different representations, including real parameters. Last, early GAs heavily relied on the bitstring representation [10, 9], and were later extended to other representations [22].

No optimization method can accurately handle *all* optimization problems, according to the "No Free Lunch Theorem" [28]. The key question thus remains to choose the good evolutionary algorithm depending on the current problem. To address this issue, comparative validations have been conducted on many test-beds, most of them involving real-valued fitness functions (e.g. the DeJong test-suite [3]); see [2, 5, 6] among many others. Except in the GA literature, little attention has been paid to boolean problems, despite the fact that real-world problems often involve discrete features. The evolutionary landscape thus appears divided: GAs are to be preferred for discrete problems only [22, 12]; In the meanwhile, the efficient strategies developed in EP and ES to evolve real-valued features seem to transpose poorly to the discrete case [1].

This paper is concerned with the mutation of discrete features in the framework of evolution strategies. The idea is to use the current population as a reservoir of information. More precisely, the worst individuals in the current population are considered as beacons signaling the limits of the present promising region, which adaptively move as the population is driven toward regions of increasingly high fitness. These beacons give rise to two mutation operators: *mutation by imitation* moves the individual at hand toward the beacons, whereas *mutation by differentiation* rather moves the individual at hand away from the beacons. These mutation operators allow individuals to exchange information and directly influence each other. This was only possible so far through recombination. The advantage of the proposed mechanism is that it allows one to control the distance between offspring and parents by fixing the number of bits to mutate, whereas classical recombination offers no guarantee as to the distance between offspring and parents.

This paper is organized as follows. Section 2 briefly reviews the main trends regarding the mutation of real-valued and discrete features in EP and ES. Mutation by imitation and mutation by differentiation are described in section 3. These operators are experimented on four problems: the Royal Road [18, 19, 8], the Ugly (GA-deceptive) problem [27], the combinatorial multiple knapsack optimization problem [21, 14], and the Long Path problem [11]. Comparative results are discussed in section 4. We conclude with some avenues for further research.

2 State of the art

The general real-valued optimization problem is stated as: $\mathcal{F}$ being a real-valued function defined on $\mathbb{R}^n$, find $x \in \mathcal{E} \subseteq \mathbb{R}^n$, such that

$$\mathcal{F}(x) = Max\{\mathcal{F}(y), y \in \mathcal{E}\}$$

Real-valued evolutionary algorithms now exist in all three communities (see for instance [16] for an example of real-valued GA). Real-valued crossover is based on the linear combination of two or several parents [16, 2], or on classical n-point recombination. Real-valued mutation involves the addition of a Gaussian random variable:

$$x_i \ := \ x_i + N(0, \sigma_i)$$

where $N(0, \sigma)$ represents the Gaussian random variable of standard deviation σ. The main difficulty remains to adjust parameters σ_i.

Most real-valued GAs use fixed σ_i, or σ_i decreasing along generations according to a fixed schedule. Early ESs use the well-known 1/5 rule [23], where a global σ is modified according to the recent effects of mutation: if more than one-fifth of the mutations have been successful in the last generation (i.e. lead to an offspring more fit than the parent), increase σ, otherwise decrease σ. The schedule for increasing and decreasing σ is geometrical schedule (the factor suggested by Schwefel [24] is 1.1). Last, in early EP [4], the standard deviation of mutation was determined on the basis of the fitness of the individual at hand, in such way that most fit individuals undergo mutation with proportionally smaller standard deviations.

Both ES and EP finally turned to *adaptive mutation*: the standard deviations σ_i are encoded in the individuals, and themselves undergo evolution. The standard deviations are therefore adjusted "for free" along evolution [2].

This mechanism encounters several difficulties in the GA framework, which uses mutation very sparingly (mutation rate per bit is often around 10^{-3}): this gives little chance for the adaptive mutation of real-valued features to actually adapt. Further, Gaussian-like mutation seems to poorly evolve discrete features [1]: when the standard deviation gets very small, mutation comes to having no effect at all on discrete features. This violates the *strong causality* principle, stating that "a small change in the parameters should result in a small change in the fitness" [23]; as a matter of fact, no "small change" below the one-bit threshold is possible in the discrete case.

3 Evolution under Influence

This section presents three population-driven mechanisms to mutate discrete features in the ES framework.

3.1 Evolution and Induction

Our approach is loosely inspired from previous work devoted to the inductive control of evolution [26, 25]: evolution is considered a sequence of events (operators giving birth to new individuals), classified good or bad depending on whether the offspring are more or less fit than the parents. These events, gathered through either spying evolution or experimenting on the current population, are processed by Inductive Learning algorithms [17, 15]. Induction can thus construct online some "law of bad events", which may then be used in a variety of ways to guide the next generations (e.g. to control the disruptiveness of evolution operators, or prevent the loss of genetic diversity). The coupling of Evolution and Induction allows one to continually create, use and update knowledge about evolution.

The present paper is similarly based on the assumption that one can get valuable hints by observing evolution, and use them to guide the further steps of evolution. The difference is that what was previously observed was the evolution operators, whereas the present work directly considers individuals.

3.2 Imitation and Differentiation

Induction needs positive and negative examples; these are respectively set to the most fit and less fit offspring [1].

Indeed, the difference of fitness between a positive and a negative example is due to the difference of their gene values. From this fact, one may infer that mutation should preserve these differences, and preferably modify those bits which take same value for the good individual at hand, and the negative examples. Practically, for each individual Ex, the profile $(p_1, \ldots, p_N)$ of its differences with the negative examples is computed, where p_i is the fraction of negative examples differing from Ex on bit x_i. *Mutation by differentiation* then mutates bit x_i in individual Ex with a probability proportional to $1 - p_i$; the resulting offspring will be still farther from the negative examples, than the parent was. The number K of bits to mutate is set by the user, and it is presently constant along evolution and over the population. Mutation by differentiation expectedly leads to highly diversified populations: if most offspring share a given bit value, this bit will be mutated with high probability. This covers as a particular case strategies devised to restore genetic diversity (see [16] among others).

One could also consider that negative examples contain valuable, if not first-rate, information: after all, these individuals or their parents have survived up to now. This leads to *mutation by imitation*, where bit x_i in individual Ex is mutated with a probability proportional to p_i. Mutation by imitation tends to "repair" explorers whenever they have lost some information shared by the previous population; it favors uniformity.

Last, and since the two above operators have opposite virtues, a third possibility consists in alternating mutation by imitation and mutation by differentiation: this hopefully allies the conservative properties of mutation by imitation and the high rate of exploration of mutation by differentiation.

We finally propose three kinds of evolution "under influence": *evolution by imitation*, *evolution by differentiation* and *alternate evolution*. The two first schemes respectively use mutation by imitation and mutation by differentiation as single evolution operator. In the third scheme, even-numbered generations undergo mutation by imitation whereas odd-numbered generations undergo mutation by differentiation.

4 Experimental Validation

Evolution under influence has been implemented in a standard $(\mu + \lambda)$ ES frame, and compared to canonical GA, canonical ES (based on the 1/5 rule), and standard adaptive ES, on four problems.

[1] Care must be exercised in discarding individuals with intermediate fitnesses: otherwise hazardous conclusions can be drawn from making distinctions between almost equally fit individuals.

4.1 The problems

• The Royal Road problem was conceived by Holland, Mitchell and Forrest [18] to study into details the interaction of features most adapted to GA search. An analysis of the unexpected difficulties of this problem is found in [19, 8].

• The Ugly problem is a GA-deceptive problem [27], built by concatenation of 10 instances of the elementary problem, defined on $\Omega = \{0,1\}^3$, by F(x) = 3 if x = 111; F(x) = 2 for x in $0\star\star$, and F(x) = 0 otherwise.

• The combinatorial multiple knapsack optimization problem [14] is defined as follows:

Given P knapsacks having respective capacities $c_1..c_P$,
Given N objects having respective costs $p_1..,p_N$,
Given the overall dimension $w_{i,j}$ of object i regarding knapsack j,
Determine a subset of objects noted $X = x_1,..x_N$, with x_i boolean, that is feasible, i.e. satisfies the constraints relative to the maximal capacities of all knapsacks, and maximizes the overall profit:

$$\text{Max } \{\sum_{i=1}^N p_i.x_i \; ; \; \forall j = 1..P, \; \sum_{i=1}^N w_{i,j} x_i < c_j.\}$$

A usual heuristics in evolutionary constrained optimization [16] consists in reducing the fitness of non feasible individuals:

$$F(X) = \begin{cases} \sum_{i=1}^N p_i x_i & \text{if } X \text{ is feasible} \\ \\ \frac{r}{2} \sum_{i=1}^N p_i x_i & \text{if } r \text{ is the percentage of satisfied constraints} \end{cases}$$

• The Long Path problem was conceived by Horn and Goldberg [11] as a unimodal problem hard for local searchers, that is, hill-climbers and the GAs mutation operator. The fitness landscape is composed of a large low-fitness plateau, within which a path of slowly increasing fitness leads to the unique optimum. This path is of length $2^{N/2}$ (N is the length of the bitstring). Two successive individuals on the path differ by one bit, while all other individuals on the path are at Hamming distance of at least 2.

4.2 Experimental settings

All results are averaged on 15 independent runs. The dynamics of evolution is visualized by plotting the (averaged) best fitness obtained for a number of fitness calculations. We compared six evolutionary schemes:

• A canonical GA (CGA) serves as reference [9]: the parents are selected based on their fitnesses with rank-based selection, 2-point crossover is applied at the rate of 0.6; the mutation is applied at the rate of $\frac{0.2}{N}$, where N denotes the length of the bitstring.

• A "traditional ES" (TES) is a boolean $(\mu + \lambda)$ evolution strategy involving a single mutation rate per bit σ ; σ is modified according to the 1/5 rule of Rechenberg [23]. The geometrical factor used to increase σ ranges from 1.1 to 2.

•An "adaptive ES" (AES) is a boolean $(\mu+\lambda)$ evolution strategy that transposes the adaptive mutation described in [1] with minor differences: Each individual is attached one mutation rate p (the probability of mutation of any single bit) which itself undergoes mutation according to Obalek's rule [20, 1]:

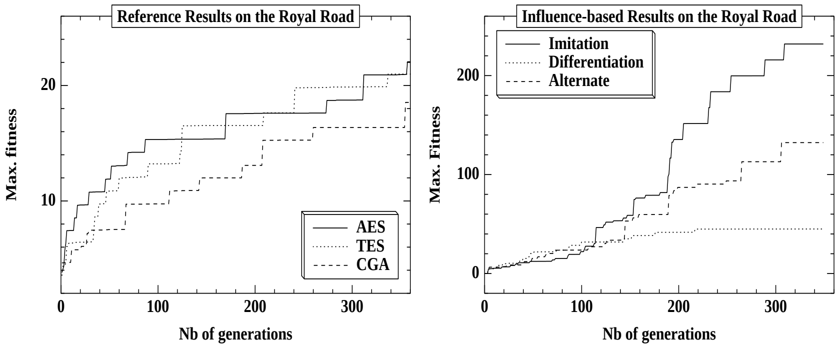
$$p := \left(1 + \frac{1-p}{p}.exp(\gamma\ .\ N(0,1))\right) \qquad \text{with } \gamma = \frac{0.5}{\sqrt{2\sqrt{N}}},$$

with a lower bound of $1/N$ on p to guarantee effective mutation.

• Mutations under influence have been implemented in the frame of a $(\mu + \lambda)$ ES. Positive and negative examples respectively are the μ most fit and μ less fit individuals among the λ offspring. Imitation- and differentiation-based evolutions respectively use mutation by imitation and mutation by differentiation as single evolution operators. Alternate evolution alternatively evolves populations through mutation by imitation and mutation by differentiation. Unless otherwise specified, the number K of bits to mutate is set to 1.

4.3 Comparative results

The Royal Road problem. The canonical GA evolves a population of 60 individuals; all other algorithms follow a $(30 + 60)$ ES scheme.



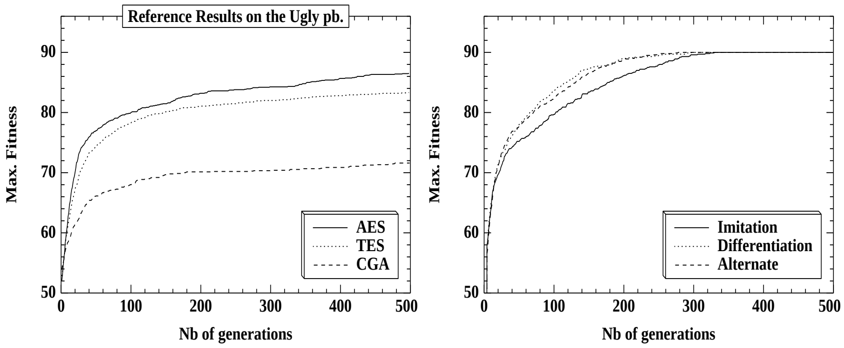
(a) Reference Results (b) Evolution under Influence

Figure 1 : The Royal Road, Dynamics of evolution (beware of the Y-scale).

The imitation scheme outstandingly outperforms all other schemes. This can be explained from the resemblance between mutation by imitation and crossover, and the fact that this problem explicitly relies on the building block hypothesis. In the meanwhile, the elitist $(\mu + \lambda)$ scheme prevents evolution from the fleeting discovery phenomenon [8], which was considered the main cause for the difficulties of standard GAs on this problem.

The Ugly problem. The canonical GA evolves a population of 40 individuals; all other algorithms follow a $(20 + 40)$ ES scheme.

The AES algorithm (ES with adaptive mutation) slightly supersedes the TES algorithm (ES following the 1/5 rule), and both significantly outperform the canonical GA algorithm on this GA-deceptive problem (Fig. 2(a)).

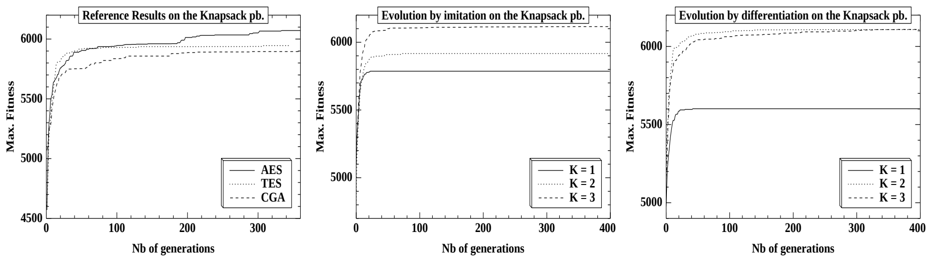


(a) Reference Results (b) Evolution under Influence

Figure 2 : The Ugly problem, Dynamics of evolution.

The differentiation scheme appears the best one on this problem, though the imitation scheme soon catches up; this can be interpreted as differentiation allows rejecting suboptimal solutions previously found (the deceptive schemata). Again, this is possible only because the $(\mu + \lambda)$ scheme counterbalances the disruptive effects of mutation by differentiation, which basically escapes any good schema the current population is settled in.

The Knapsack problem. The canonical GA evolves a population of 40 individuals; all other algorithms follow a $(20 + 40)$ ES scheme.



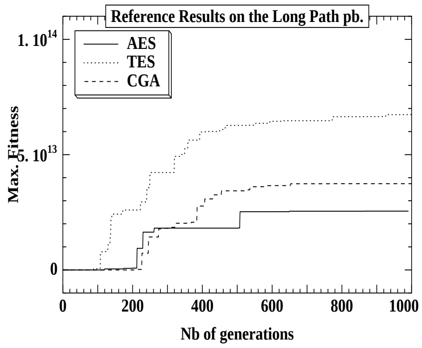
(a) Reference Results (b) Imitation Scheme (c) Diff. Scheme

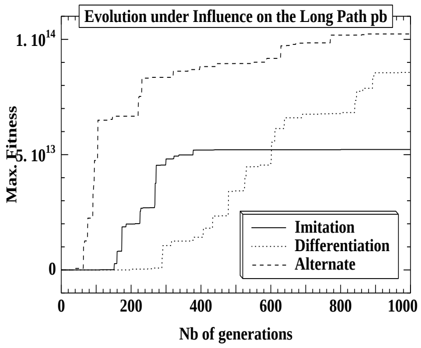
Figure 3 : The Knapsack problem, Dynamics of evolution.

This problem includes a great many of local optima, and a strong mutation is needed to avoid premature convergence; mutating more than just one bit ($K = 2$) demonstrates clearly beneficial in the Imitation and Differentiation schemes.

The Long Path problem. The canonical GA evolves a population of 21 individuals; all other algorithms follow a $(7+21)$ ES scheme. The size of the problem (number of bits of individuals) is 91. Parameter K is set to 2, the minimum jump for short-cuts on the path.

Figure 4 shows average results for 1000 generations. Note that the optimum will always be found by all methods in the (very) long run, by climbing up the path through mutations. What is measured here is the ability to find short-cuts in spite of the low-fitness plateau between different components of the path.





(a) Reference Results (b) Evolution under Influence

Figure 4 : The Long Path problem, Dynamics of evolution.

The long path problem presents intriguing particularities. First, the simple, traditional ES significantly supersedes CGA, whereas this problem was meant to be hard for mutation alone. Second, it demonstrates that alternate evolution can behave quite differently (and in fact, much better) than evolution by imitation or evolution by differentiation stand-alone. A tentative explanation is related to the accordion-like behavior of alternate evolution: mutation by imitation reduces the diversity and moves an individual toward the previous local optimum closest to this individual; then, mutation by differentiation re-increases the diversity and spreads individuals away. The combination of these two phases seemingly eases the discovery of shortcuts on the long and sneaky path toward the optimum.

5 Discussion and Perspectives

The central characteristics of evolution under influence is to allow individuals to exchange information along evolution. Such exchange is usually ensured by recombination, does it concern two parents or more [2]. And, according to the building blocks principle [10, 9], this exchange is the essential factor of biological and artificial evolution.

Another analysis [13] emphasizes that recombination can, in some contexts, be advantageously replaced by macro-mutation — equivalent to recombination with a random parent. It is noteworthy that macro-mutation and recombination offer different kinds of control on the difference between offspring and parents. Macro-mutation ensures that the offspring will be significantly different from the parent — but does not set any bias on the difference. In opposition, recombination is heavily biased: the localization, and hence the amount, of the difference between an offspring and a parent depends on the current population.

To sum up, macro-mutation controls *how much* offspring are different from parents, but does not care about *where*, that is, which bits are modified. Inversely, recombination tends to modify bits so as to copy other individuals in the population, which implies that it hardly has any effect when the population is on the verge of converging.

Mutation by imitation combines the effects of recombination and macro-mutation in that it allows for a *quantitative and qualitative control* of the difference between offspring and parents: The amount of difference (the number of mutated bits) is set by the user; The localization of the difference (which bits are modified) depends on the other individuals. Mutation by differentiation and alternate mutation similarly enable an exchange of information between individuals, that is tunable by the user.

The main originality of this work, in our opinion, is to make clear the distinction between the amount and the nature of modifications done by evolution. Further, evolution under influence determines the nature (that is, the localization) of the modifications in a way which depends on both the current population, and the individual at hand.

Many improvements can be brought to evolution under influence. Further research is concerned with adaptively adjusting the number of bits to mutate, and the kind of mutation (by imitation, differentiation, or alternate) to apply on a given individual. Mutation could also take into account, besides the difference between the current individual and the negative examples, its difference with other positive examples.

Last, this scheme will be extended to multi-modal fitness landscapes; the idea would be to evolve an individual depending on the negative examples nearest to this individual, in order to separately follow several tracks.

References

1. T. Bäck and M. Schütz. Evolution strategies for mixed-integer optimization of optical multi-layer systems. In J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, editors, *Proceedings of the 4^{th} Annual Conference on Evolutionary Programming*. MIT Press, March 1995.
2. T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.
3. K.A. DeJong. *The Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, Ann Harbor, 1975. *Dissertation Abstract International*, 36(10), 5140B. (University Microfilms No 76-9381).
4. D. B. Fogel. An analysis of evolutionary programming. In D. B. Fogel and W. Atmar, editors, *Proceedings of the 1^{st} Annual Conference on Evolutionary Programming*, pages 43–51. Evolutionary Programming Society, 1992.
5. D. B. Fogel. *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence.* IEEE Press, Piscataway, NJ, 1995.
6. D.B. Fogel and L.C. Stayton. On the effectiveness of crossover in simulated evolutionary optimization. *BioSystems*, 32:171–182, 1994.
7. L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley, 1966.
8. S. Forrest and M. Mitchell. What makes a problem hard for a genetic algorithms : Some anomalous results and their explanation. *Machine Learning*, pages 285–319, 1993.

9. D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.
10. J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.
11. J. Horn, D.E. Goldberg, and K. Deb. Long path problems. In Y. Davidor, H.-P. Schwefel, and R. Manner, editors, *Proceedings of the* 3^{rd} *Conference on Parallel Problems Solving from Nature*, page 149:158. Springer Verlag, 1994.
12. C. Z. Janikow and Z. Michalewicz. An experimental comparison of binary and floating point representations in genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of ICGA'91*, pages 31–36. Morgan Kaufmann, 1991.
13. T. Jones. Crossover, macromutation and population-based search. In L. J. Eshelman, editor, *Proceedings of ICGA'95*, pages 73–80. Morgan Kaufmann, 1995.
14. S. Khuri, T. Bäck, and J. Heitkötter. The 0/1 multiple knapsack problem and genetic algorithms. In *Proceedings of the ACM Symposium of Applied Computation*, 1994. (http:\\www-cgi.cs.cmu.edu \afs\cs.cmu.edu\projet\ai-repository).
15. Y. Kodratoff. *Introduction to Machine Learning.* Pitman Publishing, London, 1988.
16. Z. Michalewicz. *Genetic Algorithms+Data Structures=Evolution Programs.* Springer Verlag, New-York, 1996. 3rd edition.
17. R.S. Michalski. A theory and methodology of inductive learning. In R.S Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning : an artificial intelligence approach*, volume 1. Morgan Kaufmann, 1983.
18. M. Mitchell, S. Forrest, and J.H. Holland. The Royal Road for genetic algorithms : Fitness landscapes and GA performance. In F. J. Varela and P. Bourgine, editors, *Proceedings of ECAL-93*, pages 245–254. MIT Press, 1993.
19. M. Mitchell and J.H. Holland. When will a genetic algorithm outperform hill-climbing ? In S. Forrest, editor, *Proceedings of ICGA'93*, pages 647–654. Morgan Kaufmann, 1993.
20. J. Obalek. *Rekombinationsoperatoren für Evolutionsstrategieren.* PhD thesis, Universität Dortmund, Fachbereich Informatik, 1994.
21. C.C. Petersen. Computational experience with variants of the Balas algorithm applied to the selection of R & D projects. *Management Science*, 13:736–750, 1967.
22. N. J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–20, 1991.
23. I. Rechenberg. *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution.* Fromman-Holzboog Verlag, Stuttgart, 1973.
24. H.-P. Schwefel. *Numerical Optimization of Computer Models.* John Wiley & Sons, New-York, 1981. 1995 – 2^{nd} edition.
25. M. Sebag, C. Ravisé, and M. Schoenauer. Controlling evolution by means of machine learning. In L.J. Fogel, P. J. Angeline, and T. Bäck, editors, *Proceedings of the* 5^{th} *Annual Conference on Evolutionary Programming.* MIT Press, 1996.
26. M. Sebag and M. Schoenauer. Controlling crossover through inductive learning. In Y. Davidor, H.-P. Schwefel, and R. Manner, editors, *Proceedings of the* 3^{rd} *Conference on Parallel Problems Solving from Nature.* Springer-Verlag, LNCS 866, 1994.
27. D. Whitley. Fundamental principles of deception in genetic search. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms.* Morgan Kaufmann, 1991.
28. D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical report, Santa Fe Institute, 1995.