The Baldwin Effect Hinders Self-Adaptation

Jim Smith

Department of Computer Science and Creative Technologies,
University of the West of England,
Bristol, BS161QY, UK
james.smith@uwe.ac.uk
http://www.cems.uwe.ac.uk/~jsmith

Abstract. The "end-game" of evolutionary optimisation is often largely governed by the efficiency and effectiveness of searching regions of space known to contain high quality solutions. In a traditional EA this role is done via mutation, which creates a tension with its other different role of maintaining diversity. One approach to improving the efficiency of this phase is self-adaptation of the mutation rates. This leaves the fitness landscape unchanged, but adapts the shape of the probability distribution function governing the generation of new solutions. A different approach is the incorporation of local search – so-called Memetic Algorithms. Depending on the paradigm, this approach either changes the fitness landscape (Baldwinian learning) or causes a mapping to a reduced subset of the previous fitness landscape (Lamarkian learning). This paper explores the interaction between these two mechanisms. Initial results suggest that the reduction in landscape gradients brought about by the Baldwin effect can reduce the effectiveness of self-adaptation. In contrast Lamarkian learning appears to enhance the process of self-adaptation, with very different behaviours seen on different problems.

1 Introduction

Evolutionary Algorithms (EAs) are a class of population-based global search heuristics that have proved highly successful in many optimisation domains [5]. Randomised mutation and crossover operators create a non-uniform probability distribution function (pdf) over the search space for sampling new candidate solutions. The shape of this pdf is governed by a parent pool selected from the current population, the choice of recombination and mutation operators, and their associated parameters. A broader pdf allows exploration of the search space, and hence the ability to escape local optima. A narrower pdf allows exploitation of hard-won information by focussing sampling in the vicinity of promising solutions. The way in which the trade-off between these two factors is managed has a major impact on both the effectiveness and efficiency of search.

One common approach is to couple the randomised nature of EAs with a more systematic local search method to create Memetic Algorithms (MAs). This may be done in a number of ways – see e.g. [9] for a description and taxonomy and [11] for a recent survey. This paper will examine the simplest and most common:

T. Bartz-Beielstein et al. (Eds.): PPSN XIII 2014, LNCS 8672, pp. 120–129, 2014.

after recombination and mutation, each offspring undergoes local search for a specified number of iterations. In a Baldwinian paradigm, akin to life-term learning, the offspring has its fitness replaced with that of the fittest neighbour found by the local search. The Lamarkian paradigm is more drastic – both the "genome" and fitness of the offspring are replaced. Studies of these two paradigms with the EC date back to the mid-1990s. Both process alter the search landscape "seen" by the EA, but in different ways (see Section 2).

Another very common approach, is parameter adaptation. Typically an initially more uniform pdf is "narrowed" to focus more on promising regions of the search space over time. In both the combinatorial and real-valued domains, the majority of research and applications have focussed on adapting the mutation parameters [4]. Whether adaptation is driven implicitly (e.g. via self-adaption) or explicitly via the application or an "external" algorithm, a key factor is the presence of some form of evidence of the utility of an operator, or parameter value in generating high quality solutions from the current population. In the self-adaptive paradigm the evidence is implicit - successful strategies are those that produce offspring that survive, and increase their representation via association with above average quality solutions. These approaches have been successfully combined with a focus on adaptation at the memetic level [18,19], but little or no attention has been paid to the potential issues even with simple "first-generation" MAs, when the action of local search potentially destroys the link between strategies and offspring survival that is considered essential for successful self-adaptation to occur.

This paper represents a start at understanding this issue by examining the patterns of behaviour observed when applying a simple MA with self-adaptation of mutation rates to some well-understood combinatorial problems, where the "building blocks" are of different orders, so that some cannot be discovered by local search alone. Specifically it examines the following hypotheses: **H1** One-step Baldwinian learning has a "blurring" effect on the fitness landscape that reduces the effect of different mutation rates, and hence the selection pressure between them, hindering effective self-adaptation; **H2** One-step Lamarkian learning behaves differently. The mapping to a reduced search space occurring when offspring are replaced by fitter neighbours effectively increases the selection pressure towards lower mutation rates; **H3** On problems with single-bit building blocks, using multiple steps of local search compounds the effects seen above and increases the selective pressure towards lower mutation rates; **H4** In contrast, on problems with higher order building blocks, the effect of multiple steps of local search is to act as a repair function, which preserves higher mutation rates.

Section 2 provides a brief introduction to the key concepts. Section 3 describes the algorithms, test problems, and methods used to generate and analyse results. The results are presented in Section 4 and discussed in Section 5. Section 6 draws conclusions and suggests future work.

2 Background

The practice and theory of self-adaptation of mutation rates has been documented in the continuous domain since the 1960s (see, e.g., [14,3]), the binary domain since the 1990s (see [1,20,16]) and more recently for permutations [15]. A recent survey is [10]. To achieve the necessary selection pressure it has been found preferable to use a survivor:offspring ration of around1:5 which tallies with previous work in Evolution Strategies. For combinatorial problems there is evidence that the use of a continuous variable to encode for the mutation rate, and subject to log-normal adaptation, can be outperformed by a simpler scheme [21,16,17]. In this scheme the gene encoding for the mutation rate has a discrete set of alleles, and when itself subject to mutation is randomly reset with a small probability. In particular it was shown that the mechanism for adapting the encoded mutation rate is important – allowing the operator to work "on-itself" (as per [2,20]) will lead to premature convergence to sub-optimal attractors. Similar theoretical [13] and experimental (e.g. [7]) results have been found in the continuous domain. Extensive experimentation revealed that in binary search spaces different variants of self-adaptation do offer performance advantages [12], but that a deeper understanding of the processes involved is still needed.

The field of Memetic Computation encompasses a wide range of algorithms based on the concept of memes as methods for generating or improving individual solutions to one or more problem instances. Ong et. al. [11] consider a more general paradigm which uses *"the notion of meme(s) as units of information encoded in computational representations for the purposes of problem solving"*. This enticing view nevertheless requires a better understanding of the basic processes at work before more complex systems can be built. Therefore this paper is restricted to a simple first generational memetic algorithm where a greedy local search mechanism is applied to each offspring after it is created by mutation in an Evolutionary Algorithm. The number of successive neighbourhoods examined before returning to the main EA loop is controlled by a *depth* parameter.

Within a memetic algorithm, one can consider the local search stage to occur as an improvement, or developmental learning phase within the evolutionary cycle, and it is a design choice whether the changes made to the individual (*acquired traits*) should be kept in the genotype (the Lamarkian paradigm), or whether the just resulting improved fitness should be awarded to the original (pre-local search) member of the population (the Baldwin paradigm). In a classic early study, Hinton and Nowlan [8] showed that the Baldwin effect could be used to improve the evolution of artificial neural networks, and a number of researchers have studied the relative benefits of Baldwinian versus Lamarckian algorithms. These two approaches both alter the fitness landscape:

- The Baldwin effect is to replace the fitness of each point with that of its fittest neighbour. To extend the landscape metaphor, this has the effect of broadening peaks and ridges, raising the height of valleys, and generally "blurring" the landscape structure and removing gradients and fine-grained structural features in a process similar to noise removal in image processing.

- The effect of Lamarkian learning is that the fitness of points in the landscape is unchanged, but a translation occurs to the higher neighbour, so that whole swathes of low-fitness points are effectively removed from the search space.

The aim of this paper is to examine whether the impact of these two different transformations is to reduce the size of the effect of different search strategies, and hence the information available to the self-adaptation process.

3 Experimental Methodology

3.1 Algorithm

The core EA used a very standard Genetic Algorithm (GA) following the parameter values suggested by previous authors. A (100,500) selection strategy with one point crossover (with probability 0.7), and bit-flipping mutation. Local search used a Hamming neighbourhood of distance one, with a greedy pivot rule accepting the first improvement, and depths of 0,1,2 or 5 successive neighbourhoods. Note that a local search depth of 0 equates to a standard GA.

The Self-adaptation process used the scheme outlined in [16,17,21]. Each solution encodes a choice from a discrete set of values, $1.0/l * \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 1.0, 2, 5, 10\}$ where l is the length of the problem encoding. Prior to mutating the solution encoding, the gene encoding for the mutation rate is randomly reset with probability $P_{sm} = 0.1$. Although these operators and parameter values were taken as fairly standard from the literature, preliminary experimentation (not show for reasons of space) suggests that the effects observed below occur over a wide range of parameter values. One point crossover was chosen for its positional bias which matches that of the problem encodings used.

3.2 Test Functions

The first set of problems were versions of the Royal Road fitness function [6]. In these the fitness is given by the number of blocks "aligned" to the target string (all 1s) in a problem with L blocks, each of length K. To examine the effect of learning as the size of the partitions (plateaus) increased, while keeping the size of the search space the same, 60 bit problems were used with $K \in \{2, 3, 4, 5, 6, 10, 15, 20\}$. A well known property of these functions is that for $K > 1$ they possess "plateaus" of equal fitness, that represent entropic barriers to evolutionary search. Search on these problems typically proceeds via a series of "epochs". During transitions the entropy of the population is reduced as the correct alignment is found for the next block, and fixated through the population.

To understand the effect of learning on these problems, let us consider the partition of the search space corresponding to a single block. Applying one step of local search means that now K of the possible 2^K solutions in that partition now contribute to the global fitness instead of just 1. The effect of multiple steps of learning will depend on whether any of the blocks have unitation of $K-1$. The "Baldwin effect" on these landscapes is that the plateaus effectively grow in size to occupy a proportion $(K+1)/2^K$ of the partition. Regardless of mutation rate, it becomes more probable that mutation will cause a jump onto the plateau, but higher rates are more likely to destroy previously existing blocks, unless these can be repaired by multiple applications.

The effect of Lamarkian learning is subtly different - points with unitation in the partition between 0 and $K-2$ are unchanged, but those with unitation $K-1$ are removed as offspring created in those regions are moved to the single sub-solution with a unitation K. Thus the proportion of the partition corresponding to the high-fitness values is now $1/(2^k - K)$ which is smaller than the Baldwin version. Thus more of these points are at Hamming distance greater than 1, so we might expect to see the selection of higher mutation rates which are more likely to cause jumps to points at distance 1 from the optimal sub-solution.

The second class of static problems are deceptive ones, that present a fitness barrier, rather than an entropic one, to evolutionary progress to the global optimum. These so-called L "Trap" or deceptive functions of size K. This paper will consider functions composed L contiguous sub-functions. Each of these is a deceptive partition of size K bits, where the reward was $100/L$ for all 1s, otherwise $0.88 * (K - u(i))/L$ where $u(i)$ is the unitation in the i^{th} partition. Again we used 60 bit problems and the same set of values for K.

The final problem is used to explore the interaction between learning, and self-adaptation's well known ability to respond automatically to changes in the fitness landscape. Hence the third test problem used is a 200-bit variant of the unimodal OneMax function, switching to the opposite (ZeroMax) after 25 generations: Before the landscape shift, the effect of Baldwinian learning with depth d on this landscape is to assign to each genome the fitness of a individual with d more bits set to 1 - in other words the shape of the landscape is left untouched except for those few solutions with a neighourhood $H(i,j) = d$ of the global optimum, where the landscape is flat. The effect of Lamarkian search is to move each point d steps up the slope of the hill - ie. effectively to remove those points with $u(i) < d$ from the search space. In both cases the underlying structure of the problem is left unchanged, so except for the more rapid convergence to the global optimum, it is hypothesized that the self-adaptation of mutation rates will follow a similar pattern to the GA.

These values of length used were chosen to provide similar levels and speed of convergence for each problem given the selection regime and population sizes.

3.3 Methods for Analysis

Each configuration of EA without local search (GA), and with Baldwin (B) or Lamarkian (L) learning with depths 1, 2 and 5 (B-d1, ..., B-d5, L-d1 etc.) was run 100 times on each problem, with a termination criteria of 50 generations. After each generation of each run data was recorded for the best, worst and mean fitness, mean and standard deviation of mutation rates in the current population, and the total number of evaluations used.

As this paper is primarily concerned with the effect on the learning of mutation rates, algorithms are compared generation-by-generation, ignoring the fact that the local search variants make more calls to the evaluation function.

In separate experiments the mean best fitness, average evaluations to solution and success rates were compared for the seven algorithms above, and variants using a fixed mutation rate of $pm = 1/l$. Where appropriate, algorithms have been compared using statistical analysis - either at snapshots of specific generations, or averaged over the whole runs. We used SPSS v20 to conduct ANalysis of VAriance (ANOVA) followed by appropriate post-hoc testing to look for "homogenous subsets" which fail pairwise tests for statistically significant differences at the 95% confidence level. Results shown in the form $A < \{B, C\} < \{C, D\}$ mean that values for set A are significantly lower than those for sets B,C and D. Values for B are not significantly lower than those for C but are for D.

4 Results

4.1 Benchmarking Self-Adaptation

Comparing effectiveness, by pooling results and performing ANOVA on the maximum fitnesses, with the function and algorithm as independent factors showed that although there were small differences between algorithms, by 49 generations there were no statistically significant differences between fixed and self-adaptive mutation rates. Comparing the final mean mutation rates, those of the MA-B-d5 algorithm were significantly higher than the other methods, which were otherwise not significantly different.

Comparing the efficiency, as measured by when the best fitness was recorded for each run, showed that the self-adaptive variants were always faster, more significantly so with increased depth of local search. Lamarkian variants were always significantly faster than their Baldwinian counterparts and increase of depth from 0 (GA) through to 5 caused a significant increase in evaluations.

The mean best fitness results showed that there was no difference between the fixed and adaptive mutation rates for Lamarkian search, but these were always significantly better than the GA and Baldwinian MAs. In contrast, adding self-adaptation to the Baldwinian MAs significantly reduced the mean best fitness for each different depth of search.

4.2 Analysis of Evolved Behaviours on Different Functions

The next set of experiments concentrate on the effect of selection at the level of mutation rates in the presence of different forms of local search. To this end, the "strategy adaptation" parameter P_{sm} was set to 0, so each member of the initial population had its mutation rate randomly set to one of the permissible values, and offspring inherited mutation rates unchanged from their parents. The results are shown graphically in Figure 1, which shows how the patterns of the evolved behaviour change between problems and algorithms. Five characteristics of the population (best, mean, worst fitness, mean and standard deviation of mutation rate) (y-axis) are plotted against the number of generations (x-axis).

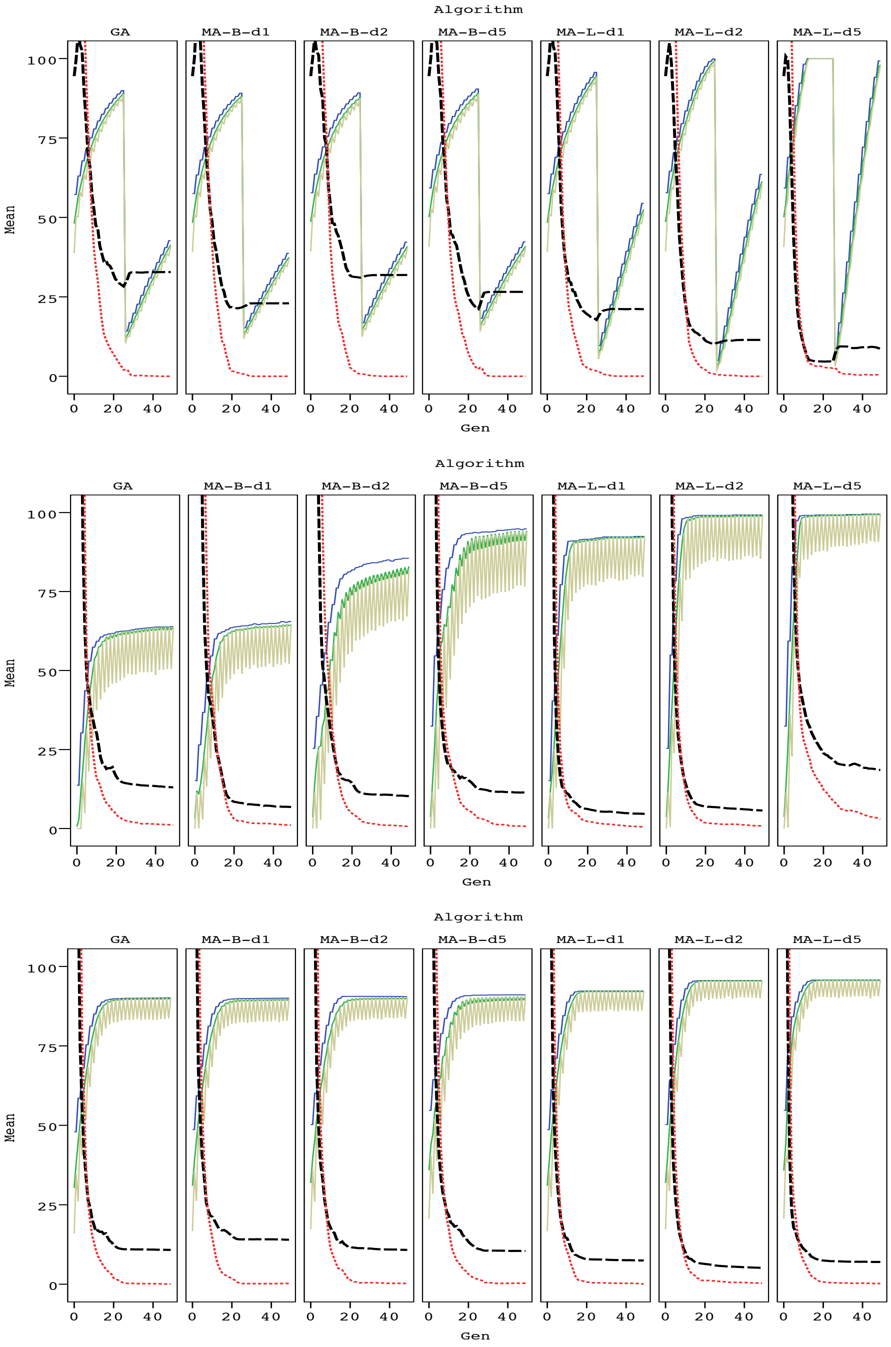

Fig. 1. Illustrative Evolving Behaviour: population best, mean and worst fitness (out of 100), mean (dashed line) and standard deviation(dotted) of mutation rates (probability x 10000). Switcher (top), Royal Road with K=8 (middle) and Trap with K=4 (bottom).

Evolution of Mutation Rates for OneMax. For both paradigms the mutation rates stabilised more slowly, and to values that decrease with increasing search depth. However for Baldwinian search, the values at generation 49 are not significantly different to the GA. The effect of selection is much more noticeable with Lamarkian learning. The mutation rates converge faster, and to lower values than the GA - not significantly so for depth 1, but the evolved rates for depths 2 and 5 are significantly different to the GA, and each other. The reduction in the standard deviation shows that this is a learned effect rather than simple drift. To confirm this, experiments were run where the function switched from OneMax to Zeromax after 25 generations. Figure 1 (top) shows a clear spike in the mutation rates after the switch. The subsequent rapid recovery in fitness, most notably for MA-L-D5, is evidence of effective self-adaption.

Results for Royal Road Functions. Figure 1 (middle) shows the evolution of behaviour on the Royal Road function with partitions of size 8. In addition to the difference in effectiveness of search, the key point to note is the consistently higher, and more varied mutation rates for Lamarkian search with depth 5, a feature that increases when the size of the sub-blocks to be optimised increased. Mutation rates also increase with depth of Baldwinian search, but the differences are not significant by generation 49

Results for Deceptive Functions. Figure 1 (bottom) shows the evolution of behaviour on the deceptive function with blocks of size 4. Note the difference in effectiveness of search. On both functions, at generation 49 the statistically homogenous subsets are, ranked according to increasing fitness; (B-d5, GA, B-d1, B-d2) $<$ L-d1 $<$ (L-d5, L-d2), where the suffix MA is omitted for brevity.

On the functions with 4-bit partitions, the Baldwin behaviour is not statistically significantly different to the GA, but there are consistently lower mutation rates for the Lamarkian learning. This difference is significant even up to generation 49 when the best value had stopped increasing. With the trap-8 function, the values are no longer statistically significant by generation 49 - but of course there are far fewer sub-functions to be optimised. Considering instead the mean mutation rates across the whole run, there is now a statistically significant difference - the values for Lamarkian learning are significantly lower than for the GA, and then in turn for the Baldwinian learning. These values reflect the speed of the adaptive process- higher mean values meaning slower adaptation.

5 Discussion

The first set of benchmarking comparisons confirmed that self-adaptation outperformed a single fixed mutation rate, as expected - working just as effectively at finding good solutions but more efficiently. Lamarkian learning improved the mean best fitness discovered. However, the interplay between the Baldwin effect and self-adaptation was not always beneficial - particular on the Royal Road landscapes where the plateaus form entropic barriers to improvement and the Baldwin effect extends those plateaus.

On the OneMax function, the hypothesis predicted that Lamarkian learning would demonstrate faster adaptation (H2) and to lower (H3) values of mutation rates than the GA. This was supported by the observations. The hypothesis H1 and H3 suggested competing effects would results from Baldwinian learning. Results confirmed that and indeed with depth 1 a slower adaption to higher rates than the GA was seen, an effect which diminished with increased local search depth, but the differences were not statistically significant by the end of even these relatively brief runs.

The results on the switcher function confirmed that self-adaptation is able to occur effectively and efficiently with Lamarkian learning up to a depth 5, possibly even suggesting a synergistic effect when compared to the GA alone.

On the Royal Road functions the hypothesised effects were not really seen except for with depth 5, where as predicted by H4, the Lamarkian search maintains higher mutation rates - which in turn lead to the continued discovery of sub-solutions. For example even after averaging over 100 runs, the middle right figure of Figure 1 shows an increase in f_{max} around 30 generations.

On the trap functions the differences are most evident in the speed of adaptation: as predicted by H1 the “blurring” effect of Baldwin learning significantly reduces the rate of adaptation to lower mutation values than the GA. In contrast, as predicted by H2, the rate of adaptation is faster for Lamarkian learning than for the GA, and hence the overall mean across all generations is lower.

6 Conclusions

This paper set out to examine the interaction between two different forms of memetic learning, and the self-adaptation of mutation rates. The primary empirical results suggest that whereas Lamarkian learning seems to reinforce self-adaptation, the Baldwin effect often hinders the process, sometimes with detrimental results on the effectiveness and efficiency of the overall search. The message of this paper is therefore perhaps unsurprising: that it is unwise to rashly mix algorithmic adaptations that work well in isolation. Clearly further studies are needed to model these effects so that the twin forces of memetics and self-adaptation can be brought to bear with reliable and predictable results.

References

1. Bäck, T.: The interaction of mutation rate, selection and self-adaptation within a genetic algorithm. In: Männer, R., Manderick, B. (eds.) Proceedings of the 2nd Conference on Parallel Problem Solving from Nature, pp. 85–94. North-Holland, Amsterdam (1992)
2. Bäck, T.: Self adaptation in genetic algorithms. In: Varela, F., Bourgine, P. (eds.) Toward a Practice of Autonomous Systems: Proceedings of the 1st European Conference on Artificial Life, pp. 263–271. MIT Press, Cambridge (1992)
3. Beyer, H.-G.: The Theory of Evolution Strategies. Springer, New York (2001)

4. Eiben, A., Michalewicz, Z., Schoenauer, M., Smith, J.: Parameter Control in Evolutionary Algorithms. In: Lobo, F.G., Lima, C.F., Michalewicz, Z. (eds.) Parameter Setting in Evolutionary Algorithms. SCI, vol. 54, pp. 19–46. Springer, Heidelberg (2007)
5. Eiben, A., Smith, J.: Introduction to Evolutionary Computation. Springer (2003)
6. Forrest, S., Mitchell, M.: Relative building block fitness and the building block hypothesis. In: Whitley, L. (ed.) Foundations of Genetic Algorithms 2, pp. 109–126. Morgan Kaufmann, San Francisco (1992)
7. Glickman, M., Sycara, K.: Reasons for premature convergence of self-adaptating mutation rates. In: 2000 Congress on Evolutionary Computation (CEC 2000), pp. 62–69. IEEE Press, Piscataway (2000)
8. Hinton, G., Nowlan, S.: How learning can guide evolution. Complex Systems 1, 495–502 (1987)
9. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: Model, taxonomy and design issues. IEEE Transactions on Evolutionary Computation 9(5), 474–488 (2005)
10. Meyer-Nieberg, S., Beyer, H.-G.: Self-Adaptation in Evolutionary Algorithms. In: Parameter setting in evolutionary algorithms, pp. 47–75. Springer, Heidelberg (2007)
11. Ong, Y.S., Lim, M.H., Chen, X.: Memetic Computation—Past, Present & Future. IEEE Computational Intelligence Magazine 5(2), 24–31 (2010)
12. Preuss, M., Bartz-Beielstein, T.: Sequential parameter optimization applied to self-adaptation for binary-coded evolutionary algorithms. In: Lobo, F.G., Lima, C.F., Michalewicz, Z. (eds.) Parameter Setting in Evolutionary Algorithms. SCI, vol. 54, pp. 91–119. Springer, Heidelberg (2007)
13. Rudolph, G.: Self-adaptive mutations lead to premature convergence. IEEE Transactions on Evolutionary Computation 5, 410–414 (2001)
14. Schwefel, H.-P.: Numerical Optimisation of Computer Models. Wiley (1981)
15. Serpell, M., Smith, J.: Self-adaption of mutation operator and probability for permutation representations in genetic algorithms. Evolutionary Computation 18(3), 491–514 (2010)
16. Smith, J.: Modelling GAs with self-adaptive mutation rates. In: Spector, L., et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), pp. 599–606. Morgan Kaufmann, San Francisco (2001)
17. Smith, J.: Parameter perturbation mechanisms in binary coded gas with self-adaptive mutation. In: Rowe, Poli, DeJong, Cotta (eds.) Foundations of Genetic Algorithms 7, pp. 329–346. Morgan Kaufmann, San Francisco (2003)
18. Smith, J.: Co-evolving memetic algorithms: A review and progress report. IEEE Transactions in Systems, Man and Cybernetics, Part B 37(1), 6–17 (2007)
19. Smith, J.: Estimating meme fitness in adaptive memetic algorithms for combinatorial problems. Evolutionary Computation 20(2), 165–188 (2012)
20. Smith, J., Fogarty, T.: Self adaptation of mutation rates in a steady state genetic algorithm. In: Proceedings of the 1996 IEEE Conference on Evolutionary Computation, pp. 318–323. IEEE Press, Piscataway (1996)
21. Stone, C., Smith, J.: Strategy parameter variety in self-adaption. In: Langdon, W., et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002), 9-13 July, pp. 586–593. Morgan Kaufmann, San Francisco (2002)