On the Interaction between Self-adaptive Mutation and Memetic Learning

J.E. Smith[1]

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 the effects of mechanisms embodying these two approaches. Initial results suggest that 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, but appropriate, 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 [7]. Much of their success comes from the use of randomised "genetic operators" –mutation and crossover – that create non-uniform probability distribution function (pdf) over the search space for generating new candidate solutions to be sampled. Given a parent pool selected from the current population (a multiset of candidate solutions), the shape of this pdf is governed by the contents of the pool, 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 evolutionary search.

One common approach is to couple the randomised nature of EAs with a more systematic local search method to create Memetic Algorithms. This may be done in a number of ways – see e.g. [15] for a description and taxonomy. This paper will examine the most straightforward and most common: after recombination and mutation, each offspring undergoes local search for a specified number of iterations. In a Baldwinian paradigm [3], 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"(representation) and fitness of the offspring are replaced. Studies of these two paradigms with the Evolutionary Computation literature date back to the mid-1990s (see e.g. [41] and other papers within that special issue), and over the last decade hundreds of papers have documented their successful application to improve the effectiveness and efficiency of evolutionary search - see e.g. [21] for a recent survey. Both process alter the search landscape "seen" by the EA, but do this in different ways – this is discussed in more depth in Section 2.3.

Another very common approach, with proven success, is to apply a method for parameter adaptation, typically with the effect that 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, since the effect of recombination lessens as the population converges. A good review may be found in [8]. 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. From early research [5] through to more current algorithms [39, 9, 16, 10], many adaptive algorithms maintain explicit archives - to record the mean improvement caused by different settings and reward the more promising. In contrast, in the self-adaptive paradigm the evidence is more implicit - successful strategies are those that produce offspring that survive, and so propagate via association. The practice and theory of self-adaptation of mutation rates has been documented in the continuous domain since [26, 4], the binary domain since [1, 2, 36, 37, 29] and for permutations [27]. More recent surveys may be seen in [33, 19].

These approaches have been successfully combined in, for example, the COMA framework [30, 32, 28, 34, 35], but much of that work focussed on the issues adaptation at the memetic level to create what Meuth *et. al.* called "second and third generation MAs" [18]. So far no attention that the author is aware of 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 building an understanding of this issue by examining the patterns of behaviour observed when applying a simple memetic algorithm with self-adaptation of mutation rates to some well-understood combinatorial problems, where the "building blocks" to be found and propagated are of different orders, so that some cannot be discovered by local search alone. Specifically it examines the following hypotheses:

[1] Department of Computer Science and Creative Technologies, University of the West of England, Bristol, UK. email: james.smith@uwe.ac.uk

- H1: That one-step Baldwinian learning has a blurring effect on the fitness landscape which reduces the selection pressure on different mutation rates, slowing the process of self-adaptation.
- H2: That one-step Lamarkian learning behaves differently - the mapping to a reduced search space that occurs when offspring are replaced by fitter neighbours effectively increases the selection pressure towards lower mutation rates.
- H3: That 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, allowing the preservation of the higher mutation rates needed to discover the "optimal" building blocks.

The paper is set out as follows. Section 2 provides a (necessarily) brief introduction to the key concepts of self-adaptation and Lamarkian search and the Baldwin effect. 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 before Section 6 draws conclusions and suggests future work.

2 Background

2.1 Self-Adaptation of Mutation Rates

The practice of using adaptive mechanisms to alter operator choices and parameters has attracted much attention. However the space of operators and parameters is large, and the mapping to the resulting quality of solution found is complex, still not well-understood, and problem dependent. Therefore hand-designed mechanisms have had relatively less success, and there has been natural interest in the application of evolutionary algorithms to search this space. In particular the use of "Self-Adaptation", where the operator's parameters are encoded within the individuals and subjected to evolution was established in the continuous domain within Evolution Strategies [26], and Stephens et al. have shown in general that adding self-adaptive genes to encodings can create evolutionary advantages [37].

Bäck's work in self-adapting the mutation rate to use for binary encodings within generational GAs [1, 2] proved the concept, and established the need for appropriate selection pressure. Smith and Fogarty examined the encoding and conditions necessary to translate this to a steady-state GA [36]. To achieve the necessary selection pressure they employed a cloning mechanism: from the single offspring resulting from crossover they derived a set of clones. The mutation rate of each clone was then modified with a certain probability, before being applied. The fittest resulting solution was then returned to the population. Results showed that the number of clones that led to the shortest tour length was 5 and that this tallied with previous work in Evolution Strategies (where the ratio of μ to λ is typically in the range $5-7$) and Bäck's implementation of truncation selection. They found that adding a self-adaptive mutation rate improved the performance of the GA and removed a non-trivial parameter from the GA. They also examined a variety of different ways of encoding the mutation parameter.

In subsequent work, Stone and Smith showed that for combinatorial problems the use of a continuous variable to encode for the mutation rate, subject to log-normal adaptation was outperformed by a simpler scheme [38]. In their method the value of the gene encoding for the mutation rate had a discrete set of alleles i.e. the mutation rate came from a fixed set, and when subject to mutation was randomly reset with a small probability. This has been examined experimentally and theoretically in [29] and [31]. In particular it was shown that the way that the encoded mutation rate is perturbed is important – allowing the operator to work "on-itself" (as per [2, 36]) will lead to premature convergence to sub-optimal attractors. This effect has subsequently been rediscovered elsewhere [24]. Similar results have been found in the continuous domain experimentally (e.g. [12]) and theoretically [25].

Extensive experimental studies using Sequential Parameter Optimisation have revealed that in binary search spaces different variants of self-adaptation do offer performance advantages [23]. However the very mixed results clearly indicate the need for a deeper understanding of the processes involved.

2.2 Memetic Algorithms

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. Rather than just local search-evolutionary hybrids, [21] consider Memetic Computation as 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"*. In their more general view memes might be represented as *"decision trees, artificial neural networks, fuzzy system, graphs etc"* , and are not necessarily coupled to any evolutionary components at all, requiring simply a method for credit assignment. This enticing view offers the promise of memes capturing useful structural and behavioural patterns which can be carried between instances of the same problem, as is being explored in e.g. [40].

This paper is restricted to the broad class of "Memetic Algorithms" (MAs). Introduced by Moscato [20], these combine population-based global search heuristics (such as EAs) with heuristics that attempt to improve a single solution. Meuth et al. [18] distinguish between:

- First Generation MAs - which they define as *"Global search paired with local search"*,
- Second Generation MAs - *"Global search with multiple local optimizers. Memetic information (Choice of optimizer) passed to offspring (Lamarckian evolution)"*,
- Third Generation MAs: - *"Global search with multiple local optimizers. Memetic information (Choice of local optimizer) passed to offspring (Lamarckian Evolution). A mapping between evolutionary trajectory and choice of local optimizer is learned"*.

Although powerful paradigms with increasing number of successful applications, adaptive second and third generation MAs bring with them a number of issues concerning the credit-assignment problem [22, 35]. For these reasons this initial study concentrates on a simple first generational memetic algorithm based on a greedy bit-flipping mechanism. For a binary string with length l the Hamming distance between two strings as $H(i,j) = \sum_{k=1}^{l} \mid j[k] - i[k] \mid$, so this algorithm uses a neighbourhood function $n_{H1}(i) = j : H(i,j) = 1$. This local search algorithm can be illustrated by the pseudocode given in Fig. 1.

2.3 Lamarckianism and the Baldwin Effect

The framework of the local search algorithm outlined above works on the assumption that the current incumbent solution is always replaced by the fitter neighbour when found. Within a memetic algo-

```
Bit-Flipping Local Search:
Begin
  /* given a starting solution $i$ */
  /* and the neighbourhood function $n_{H1}(i)$ */
  set $best = i$;
  set $iterations = 0$;
  Repeat Until ( depth condition is satisfied )
  Do
    set $count = 0$;
    Repeat Until ( pivot rule is satisfied )
    Do
      generate the next neighbour $j \in n_{H1}(best)$;
      set $count = count + 1$;
      If ($f(j)$ is better than $f(best)$) Then
        set $best = j$;
      Fi
    Od
    set $iterations = iterations + 1$;
    set $f(i) = f(best)$;
    If (Lamarkian) Then
      set $i = best$;
    Fi
  Od
End.
```

Figure 1: Pseudocode of a local search algorithm

rithm, one can consider the local search stage to occur as an improvement, or developmental learning phase within the evolutionary cycle, and (taking our cue from biology) one should consider whether the changes made to the individual (*acquired traits*) should be kept in the genotype, or whether the resulting improved fitness should be awarded to the original (pre-local search) member of the population.

The question of whether acquired traits could be inherited by an individual's offspring was a major issue in nineteenth century, with Lamarck arguing in favour, whereas the Baldwin effect [3] suggests a mechanism whereby evolutionary progress can be guided towards favourable adaptation without the changes in individuals' fitness arising from learning or development being reflected in changed genetic characteristics. Modern theories of genetics strongly favour the latter viewpoint.

Luckily, computer algorithms are not restricted by these biological constraints, and so in practice both schemes are usually possible to implement within a memetic algorithm. In general MAs are referred to as Lamarckian if the result of the local search stage replaces the individual in the population, and Baldwinian if the original member is kept, but has as its fitness the value belonging to the outcome of the local search process. In a classic early study, Hinton and Nowlan [13] 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 [14, 17, 41, 44, 45]. In practice most recent work has tended to use either a pure Lamarckian approach, or a probabilistic combination of the two approaches, such that the improved fitness is always used, and the improved individual replaces the original with a given probability.

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

3 Experimental Methodology

3.1 Algorithm

The core Evolutionary Algorithm used a very standard Genetic Algorithm (GA) with a (100,500) model for population management and the following operators and parameters.

Table 1: EA operators and parameters

Operator/Parameter	Value
μ	100
λ	500
Representation	Binary Strings
Solution Length	Problem-dependant
Crossover	One-Point
Crossover Probability	0.7
Mutation Operator	Bit-Flipping
Mutation Rate Encoding	Integer in range [1,10]
Mutation Strategy Rate P_{sm}	0.05 or 0
Parent Selection	Tournament Size 1
Survivor Selection	Truncation
Local Search Neighbourhood	Hamming Distance 1
Local search pivot rule	Greedy
Depth of local search d	1, 2 or 5 iterations

The Self-adaptation process used the scheme outlined in [29, 31, 38]. Rather than attempting to adapt a continuous mutation rate parameter, each solution encodes a choice from a discrete set of values, $1.0/l * \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 1.0, 2, MIN(0.25 * l, 5.0), MIN(0.25 * l, 10.0)\}$ 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}.

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 encoding used for the Royal Road and Trap functions.

3.2 Test Functions

The first test problem used is a 200-bit version of the unimodal One-Max function:

$$f(i) = u(i) * 100/l. \qquad (1)$$

where the unitation $u(i) = \sum_{j=1}^{l} i[j]$.

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.

The second two problems used were 64-bit versions of the type R1 Royal Road fitness function [11] where the fitness is given by the number of blocks "aligned" to the target string in a problem with L blocks, each of length K. Assuming (without loss of generality) that the target string is all 1s for each block:

$$f_{R1}(i) = \sum_{l=1}^{L} \Pi_{j \in block_l} \; i_j \qquad (2)$$

A well known property of these functions is that for $K > 1$ they possess "plateaus" of equal fitness, and as discussed in [42] these represent entropic barriers to evolutionary search. As has been extensively documented, search on these problems typically proceeds via a series of "epochs", and 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 examine the effect of learning two different 64-bit versions with different sized plateaus were used - denoted hereafter as "Royal-4bit" ($L = 16, K = 4$) and "Royal-8bit" ($L = K = 8$).

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 move 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 preferential selection for higher mutation rates which are more likely to cause jumps to points at 1-change remove from the optimal sub-solution.

A third class of problems which present a fitness barrier, rather than an entropic one to evolutionary progress to the global optimum is characterised by so-called L "Trap" or deceptive functions of size K. This paper will consider functions with a deceptive partition whose fitness varies as a function of the unitation $u_j(i)$ in the $j - th$ partition as given in [6]:

$$f(i) = 100.0/L * \sum_{j=1}^{L} \begin{cases} a(K - 1 - u_j(i)) & u_j(i) \leq K \\ 1 & otherwise \end{cases} \qquad (3)$$

Experiments used a 80 bit four-trap problem and a 64 bit 8 trap problem - so L = 20 and 8 respectively, and the constant a at values 0.1 for $K = 8$ and 0.2 for $K = 4$ to ensure that the optimal configuration for the partition outscores the deceptive optima.

For each problem, these values of length used were chosen to provide similar levels and speed of convergence to the other problems 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 mostly compared on a generation-by-generation basis, ignoring the fact that the local search variants make more calls to the evaluation function. Results are displayed graphically, and where claims are made about values at the end of runs, these are confirmed by analysis of the results at generation 49 using Analysis of Variance followed by post-hoc testing using Tukeys HSD test at the 95% confidence level. Statistical analysis was carried out using the SPSS v. 20 software.

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 $pm = 1/l$. These were repeated with different selection regimes - namely (100,500) with uniform parent selection, and (500,500) with tournaments of size 5 to select parents. The latter should provide similar selection pressure (according to takeover times) and has been suggested to outperform truncation selection [24].

4 Results

4.1 Benchmarking Self-Adaptation

Performing an Analysis of Variance (ANOVA) on the (100,500) results for maximum fitness after 49 generations, with the function, and algorithm as independent factors showed that although there were small differences between algorithms, by that stage there were not 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. The overall picture was not different with the (500,500) regime.

Comparing the efficiency, as measured by when the best fitness was recorded for each run, ANOVA with this as the dependant variable, and function and algorithm as independent variables showed that the self-adaptive variants were always faster, more significantly so with increased depth of local search. Lamarkian variants were always significantly was faster than their Baldwinian counterparts and each step of increasing depth from 0 (GA) through 1,2 and then 5 caused a significant increase in evaluations.

The mean best fitness results for (100,500) 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.

In order to examine the interplay of effects further, 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 from their parents, but there was no further perturbation of these strategy parameters.

4.2 Evolution of Mutation Rates for OneMax

Figure 2 shows the evolution of the fitness values in the population, the mean and standard deviation of encoded mutation rates on the OneMax function. For both learning paradigms the rates stabilise 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.

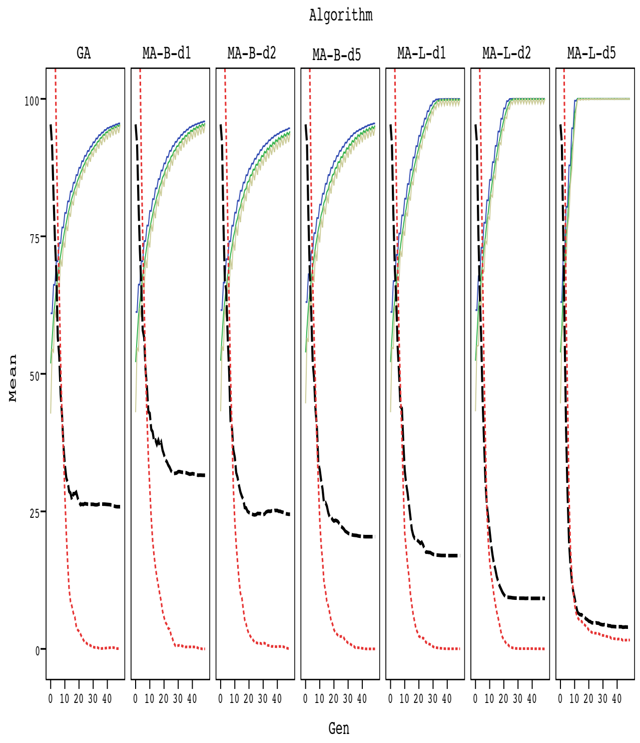

Figure 2: Evolving Behaviour on OneMax. Increasing lines are best,mean and worst fitness in population. Dashed and dotted lines are mean and standard deviation of mutation rates.

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 3 shows the evolved behaviour on this “switching” problem. The clear spike in the evolved mutation rates after the switch, and subsequent rapid recovery in fitness values, most notably for MA-L-D5 demonstrates that effective self-adapation is occurring.

4.3 Results for Royal Road Functions

Figure 4 shows the evolution of behaviour on the Royal Road function with different sized blocks. 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 statistically significant by generation 49

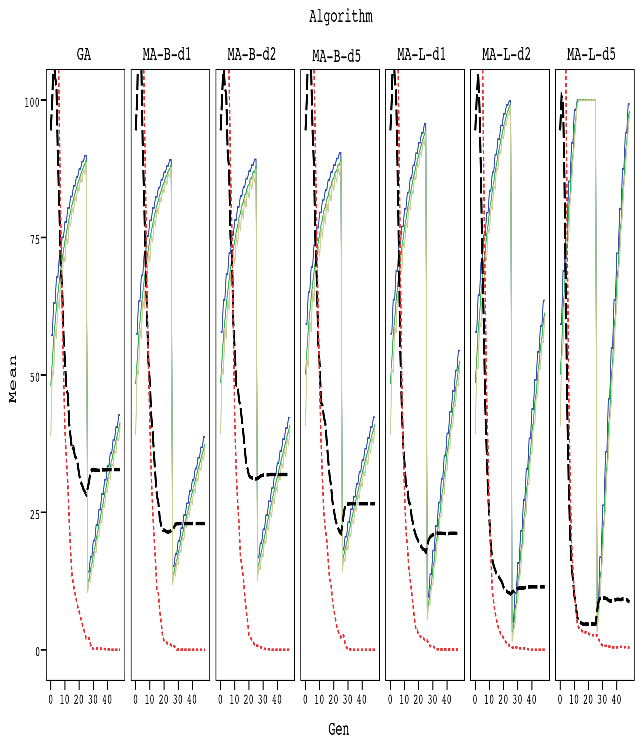

Figure 3: Evolving Behaviour on Switching function. Increasing lines are best,mean and worst fitness in population. Dashed and dotted lines are mean and standard deviation of mutation rates.

4.4 Results for Deceptive Functions

Figure 5 shows the evolution of behaviour on the deceptive function with different sized blocks: Trap-4 and Trap-8. 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 subfunctions 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

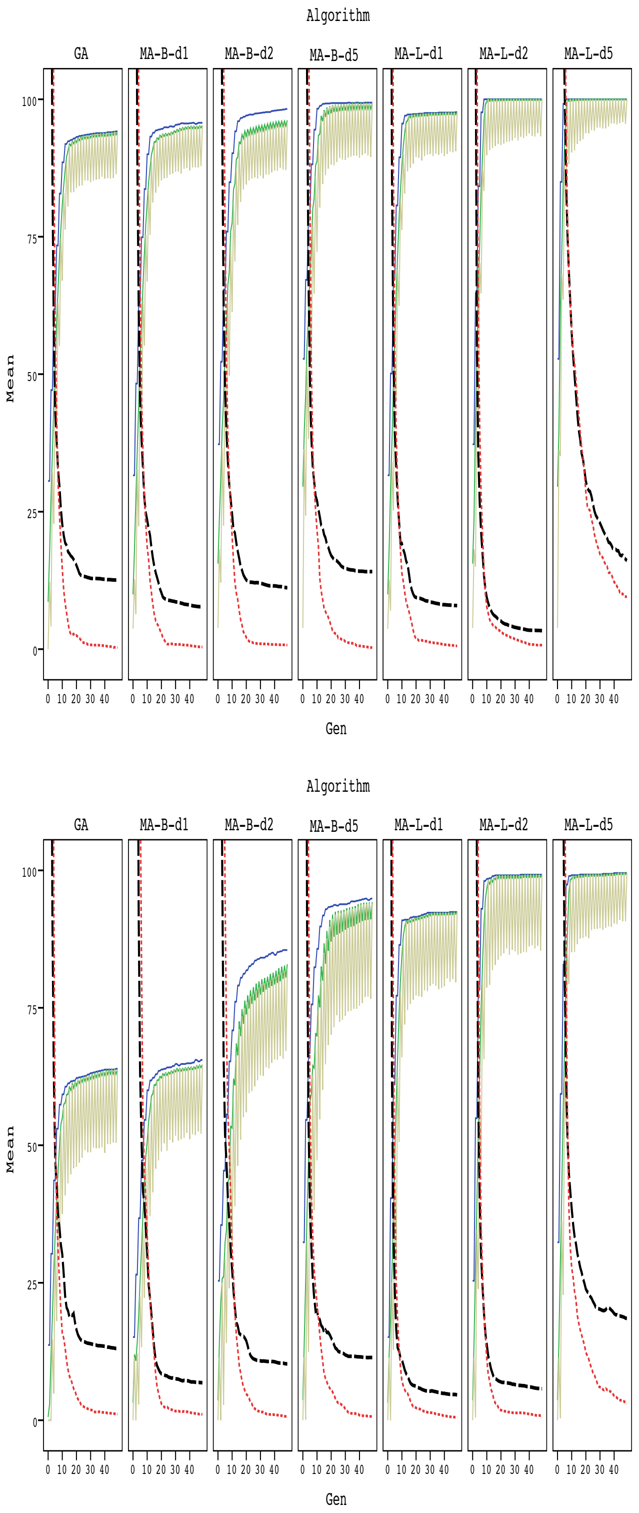

Figure 4: Evolving Behaviour on Royal Road functions with blocks of size 4 (top) and 8 (bottom). Increasing lines are best,mean and worst fitness in population. Dashed and dotted lines are mean and standard deviation of mutation rates.

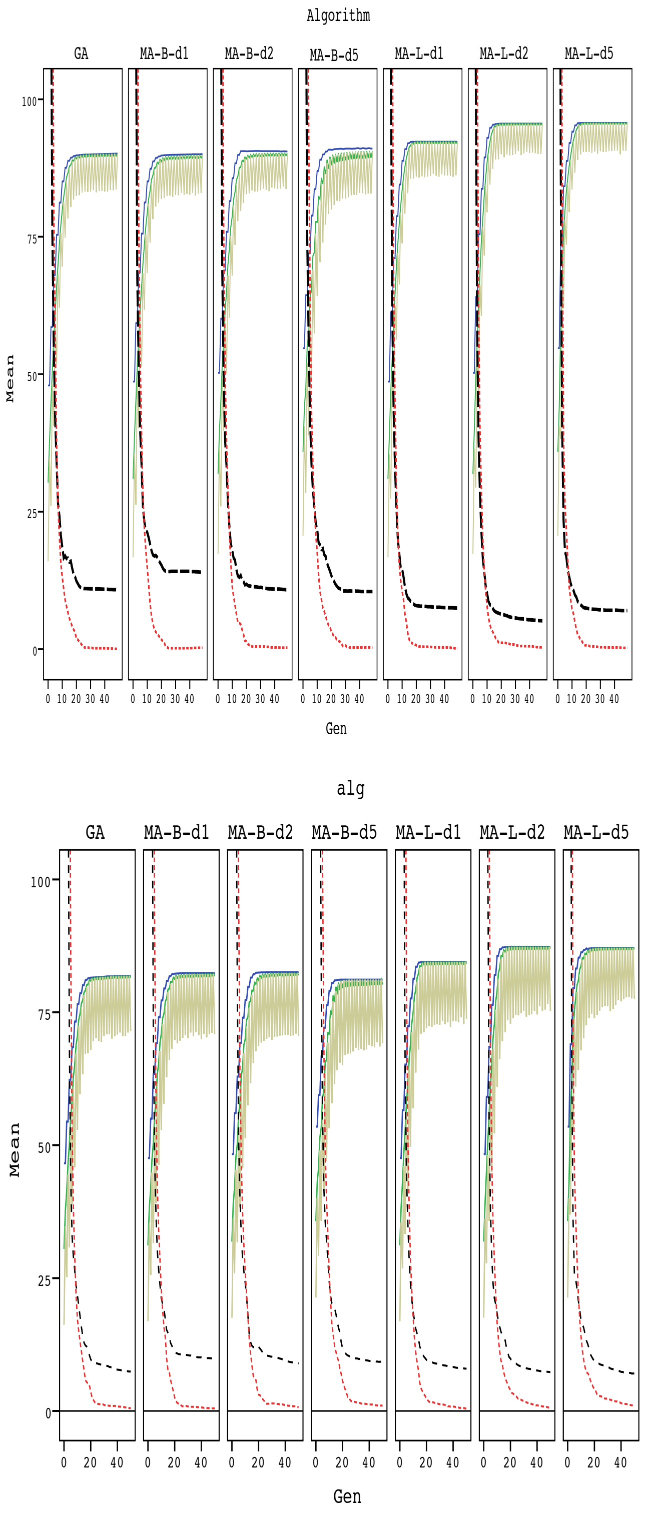

Figure 5: Evolving Behaviour on Deceptive functions with blocks of size 4 (top) and 8 (bottom). Increasing lines are best,mean and worst fitness in population. Dashed and dotted lines are mean and standard deviation of mutation rates.

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 bottom right figure of Figure 4 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. A number of hypotheses were proposed to describe the effects. Results suggest that there are indeed significant interactions, but the hypothesis themselves require significant clarification as in cases they work against each other.

The primary empirical results suggest that whereas Lamarkian learning seems to reinforce the self-adaptation process, the Baldwin effect often hinders the process, sometimes with detrimental results on the effectiveness and efficiency of the overall search process.

The primary message of this paper is therefore perhaps unsurprising: that it is unwise to rashly mix algorithmic adaptations that work well in isolation. Clearly further empirical and theoretical 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] T. Bäck, 'The interaction of mutation rate, selection and self-adaptation within a genetic algorithm', in *Proceedings of the 2nd Conference on Parallel Problem Solving from Nature*, eds., R. Männer and B. Manderick, pp. 85–94. North-Holland, Amsterdam, (1992).

[2] T. Bäck, 'Self adaptation in genetic algorithms', in *Toward a Practice of Autonomous Systems: Proceedings of the 1st European Conference on Artificial Life*, eds., F.J. Varela and P. Bourgine, pp. 263–271. MIT Press, Cambridge, MA, (1992).

[3] J.M. Baldwin, 'A new factor in evolution', *American Naturalist*, **30**, (1896).

[4] H.-G. Beyer, *The Theory of Evolution Strategies*, Springer, Berlin, Heidelberg, New York, 2001.

[5] L. Davis, 'Adapting operator probabilities in genetic algorithms', in *Proceedings of the 3rd International Conference on Genetic Algorithms*, ed., J.D. Schaffer, pp. 61–69. Morgan Kaufmann, San Francisco, (1989).

[6] K. Deb and D.E. Goldberg, 'Analyzing deception in trap functions', In Whitley [43], pp. 93–108.

[7] A.E. Eiben and J.E. Smith, *Introduction to Evolutionary Computation*, Springer, 2003.

[8] Aguston Eiben, Zbigniew Michalewicz, Marc Schoenauer, and Jim Smith, 'Parameter Control in Evolutionary Algorithms', in *Parameter Setting in Evolutionary Algorithms*, eds., Fernando G. Lobo, Cláudio F. Lima, and Zbigniew Michalewicz, volume 54 of *Studies in Computational Intelligence*, 19–46, Springer Verlag, (2007).

[9] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag, 'Extreme Value Based Adaptive Operator Selection', in *Parallel Problem Solving from Nature (PPSN X)*, ed., Parallel Problem Solving from Nature (PPSN X) ed, volume 5199 of *LNCS*, pp. 175–184. Springer, (September 2008).

[10] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag, 'Analyzing bandit-based adaptive operator selection mechanisms', *Annals of Mathematics and Artificial Intelligence – Special Issue on Learning and Intelligent Optimization*, (September 2010).

[11] S. Forrest and M. Mitchell, 'Relative building block fitness and the building block hypothesis', In Whitley [43], pp. 109–126.

[12] M. Glickman and K. Sycara, 'Reasons for premature convergence of self-adaptating mutation rates', in *2000 Congress on Evolutionary Computation (CEC'2000)*, pp. 62–69. IEEE Press, Piscataway, NJ, (2000).

[13] G.E. Hinton and S.J. Nowlan, 'How learning can guide evolution', *Complex Systems*, **1**, 495–502, (1987).

[14] C.R. Houck, J.A. Joines, M.G. Kay, and J.R. Wilson, 'Empirical investigation of the benefits of partial Lamarckianism', *Evolutionary Computation*, **5**(1), 31–60, (1997).

[15] N. Krasnogor and J.E. Smith, 'A tutorial for competent memetic algorithms: Model, taxonomy and design issues', *IEEE Transactions on Evolutionary Computation*, **9**(5), 474–488, (2005).

[16] Jorge Maturana, Álvaro Fialho, Frédéric Saubion, Marc Schoenauer, and Michèle Sebag, 'Extreme Compass and Dynamic Multi-Armed Bandits for Adaptive Operator Selection', in *IEEE Congress on Evolutionary Computation*, Trondheim Norvège, (2009).

[17] G. Mayley, 'Landscapes, learning costs and genetic assimilation', *Evolutionary Computation*, **4**(3), 213–234, (1996).

[18] R Meuth, MH Lim, YS Ong, and DC Wunsch, 'A proposition on memes and meta-memes in computing for higher-order learning', *Memetic Computing*, **1**(2), 85–100, (2009).

[19] Silja Meyer-Nieberg and Hans-Georg Beyer, 'Self-Adaptation in Evolutionary Algorithms', in *Parameter setting in evolutionary algorithms*, 47–75, Springer Berlin Heidelberg, Berlin, Heidelberg, (2007).

[20] P.A. Moscato, 'On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms', Technical Report Caltech Concurrent Computation Program Report 826, Caltech, Caltech, Pasadena, California, (1989).

[21] Y S Ong, M H Lim, and X Chen, 'Memetic Computation—Past, Present & Future [Research Frontier]', *Computational Intelligence Magazine, IEEE*, **5**(2), 24–31, (2010).

[22] Y.S. Ong, M.H. Lim, N. Zhu, and K.W. Wong, 'Classification of adaptive memetic algorithms: A comparative study', *IEEE Transactions on Systems Man and Cybernetics Part B*, **36**(1), (2006).

[23] M Preuss and T Bartz-Beielstein, 'Sequential parameter optimization applied to self-adaptation for binary-coded evolutionary algorithms', in *Parameter Setting in Evolutionary Algorithms*, eds., Fernando G. Lobo, Cláudio F. Lima, and Zbigniew Michalewicz, volume 54 of *Studies in Computational Intelligence*, 91–119, Springer Verlag, (2007).

[24] William Rand and Rick Riolo, 'The problem with a self-adaptative mutation rate in some environments: a case study using the shaky ladder hyperplane-defined functions', in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. ACM Request Permissions, (June 2005).

[25] G. Rudolph, 'Self-adaptive mutations may lead to premature convergence', *IEEE Transactions on Evolutionary Computation*, **5**, 410–414, (August 2001).

[26] H.-P. Schwefel, *Numerical Optimisation of Computer Models*, Wiley, New York, 1981.

[27] M. Serpell and J.E. Smith, 'Self-adaption of mutation operator and probability for permutation representations in genetic algorithms', *Evolutionary Computation*, **18**(3), 491–514, (2010).

[28] J. Smith, 'Credit assignment in adaptive memetic algorithms', in *Pro-*

ceedings of Gecco, the ACM-SIGEVO conference on Evolutionary Computation, pp. 1412–1419, (2007).

[29] J.E. Smith, 'Modelling GAs with self-adaptive mutation rates', in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, eds., L. Spector, E. Goodman, A. Wu, W.B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke, pp. 599–606. Morgan Kaufmann, San Francisco, (2001).

[30] J.E. Smith, 'Co-evolution of memetic algorithms: Initial investigations', in *Proceedings of the 7th Conference on Parallel Problem Solving from Nature*, eds., J.J. Merelo Guervos, P. Adamidis, H.-G. Beyer, J.-L. Fernandez-Villacanas, and H.-P. Schwefel, number 2439 in Lecture Notes in Computer Science, pp. 537–548. Springer, Berlin, Heidelberg, New York, (2002).

[31] J.E. Smith, 'Parameter perturbation mechanisms in binary coded gas with self-adaptive mutation', in *Foundations of Genetic Algorithms 7*, eds., Rowe, Poli, DeJong, and Cotta, pp. 329–346. Morgan Kaufmann, San Francisco, (2003).

[32] J.E. Smith, 'Co-evolving memetic algorithms: A review and progress report', *IEEE Transactions in Systems, Man and Cybernetics, part B*, **37**(1), 6–17, (2007).

[33] J.E. Smith, 'Self-adaptation in evolutionary algorithms for combinatorial optimisation', in *Adaptive and Multilevel Metaheuristics*, eds., C. Cotta, M. Sevaux, and K. Sorenson, 31–57, Springer, Berlin, Heidelberg, New York, (2008).

[34] J.E. Smith, 'Meme fitness and memepool sizes in coevolutionary memetic algorithms', in *Proceedings 2010 World Conference on Computational Intelligence*, pp. 1–8. IEEE Press, (2010).

[35] J.E. Smith", 'Estimating meme fitness in adaptive memetic algorithms for combinatorial problems', *Evolutionary Computation*, **20**(2), 165–188, (2012).

[36] J.E. Smith and T.C. Fogarty, '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, NJ, (1996).

[37] C.R. Stephens, I. Garcia Olmedo, J. Moro Vargas, and H. Waelbroeck, 'Self-adaptation in evolving systems', *Artificial life*, **4**, 183–201, (1998).

[38] C. Stone and J.E. Smith, 'Strategy parameter variety in self-adaption', in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, eds., W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, pp. 586–593. Morgan Kaufmann, San Francisco, (9-13 July 2002).

[39] D. Thierens, 'Adaptive strategies for operator allocation', in *Parameter Setting in Evolutionary Algorithms*, eds., F.G. Lobo, C.F. Lima, and Z. Michalewicz, 77–90, Springer, Berlin, (2007).

[40] C.K. Ting, W.M. Zeng, and T.C.Lin, 'Linkage discovery through data mining', *Computational Intelligence Magazine*, **5**, 10–13, (2010).

[41] P.D. Turney, 'How to shift bias: lessons from the Baldwin effect', *Evolutionary Computation*, **4**(3), 271–295, (1996).

[42] E. van Nimwegen, *The Statistical Dynamics of Epochal Evolution*, Ph.D. dissertation, Utrecht University and Santa Fe Institute, 1999.

[43] L.D. Whitley, ed. *Foundations of Genetic Algorithms - 2*. Morgan Kaufmann, San Francisco, 1993.

[44] L.D. Whitley, S. Gordon, and K.E. Mathias, 'Lamarkian evolution, the Baldwin effect, and function optimisation', in *Proceedings of the 3rd Conference on Parallel Problem Solving from Nature*, eds., Y. Davidor, H.-P. Schwefel, and R. Männer, number 866 in Lecture Notes in Computer Science, pp. 6–15. Springer, Berlin, Heidelberg, New York, (1994).

[45] L.D. Whitley and F. Gruau, 'Adding learning to the cellular development of neural networks: evolution and the Baldwin effect', *Evolutionary Computation*, **1**, 213–233, (1993).