Estimating Meme Fitness in Adaptive Memetic Algorithms for Combinatorial Problems

J. E. Smith james.smith@uwe.ac.uk
Department of Computer Science and Creative Technologies,
University of the West of England, BS16 1QY, U.K.

Abstract

Among the most promising and active research areas in heuristic optimisation is the field of adaptive memetic algorithms (AMAs). These gain much of their reported robustness by adapting the probability with which each of a set of local improvement operators is applied, according to an estimate of their current value to the search process. This paper addresses the issue of how the current value should be estimated. Assuming the estimate occurs over several applications of a meme, we consider whether the extreme or mean improvements should be used, and whether this aggregation should be global, or local to some part of the solution space. To investigate these issues, we use the well-established COMA framework that coevolves the specification of a population of memes (representing different local search algorithms) alongside a population of candidate solutions to the problem at hand. Two very different memetic algorithms are considered: the first using adaptive operator pursuit to adjust the probabilities of applying a fixed set of memes, and a second which applies genetic operators to dynamically adapt and create memes and their functional definitions. For the latter, especially on combinatorial problems, credit assignment mechanisms based on historical records, or on notions of landscape locality, will have limited application, and it is necessary to estimate the value of a meme via some form of sampling. The results on a set of binary encoded combinatorial problems show that both methods are very effective, and that for some problems it is necessary to use thousands of variables in order to tease apart the differences between different reward schemes. However, for both memetic algorithms, a significant pattern emerges that reward based on mean improvement is better than that based on extreme improvement. This contradicts recent findings from adapting the parameters of operators involved in global evolutionary search. The results also show that local reward schemes outperform global reward schemes in combinatorial spaces, unlike in continuous spaces. An analysis of evolving meme behaviour is used to explain these findings.

1 Introduction

Among the most promising and active research areas in heuristic optimisation is the field of adaptive memetic algorithms (AMAs). These couple global search, via an evolutionary algorithm (EA), with solution improvement via the selective application of one or more memes representing local search strategies. AMAs gain much of their reported robustness by adapting the probability with which each of a set of local improvement operators is applied, according to an estimate of their current value to the search process. The concept of assigning credit to different memes, based in some way on their perceived current benefit, is closely linked to a long, and ongoing, history of operator and parameter adaptation within EAs and related methods such as hyperheuristics. The key issues from that research may be summarised as the evidence used by an operator

or parameter adaptation algorithm, the scope of that algorithm (e.g., whether it makes global changes or attempts to model the local landscape), and the mechanism used to calculate new values according to the evidence.

Based on insights from that field, Ong et al. (2006) created a taxonomy of early work in AMAs and provided some useful benchmark results in continuous search spaces, where notions of locality are implicit and the link between global and local behaviours is fairly transparent. Tellingly, they reported a preference for global adaptation, which contrasts with the local landscape modelling of state of the art EAs such as CMA-ES (Ostermeier et al., 1994). Chen et al. (2011) report a large number of papers in which AMAs have delivered highly impressive results across a range of application domains. However, the vast majority of researchers have considered what has been called second generation MAs (2GMAs),[1] where the set of memes available is fixed a priori, so that historical records of operator utility are available and relevant. Specific evidence has been drawn from historical records in a number of different ways, and a wide range of mechanisms has been used to adapt meme choice, often based on domain-specific insights. As concluded by Ong et al. (2006), there is still a lack of general understanding of the issues surrounding credit assignment in 2GMAs.

Moreover, Ong's findings, and many of the subsequent algorithm designs, rely implicitly on the properties of continuous spaces. This could be via the use of an estimated gradient to inform local search, or through the notion of a shared landscape structure to estimate the likely rates of progress of global and local search operators, and hence to adapt the ratio of computational effort allotted to global/local search. While these avenues have yielded valuable insights, as we have argued elsewhere (see, e.g., Stone and Smith, 2002; or Serpell and Smith, 2010), the situation regarding adaptation is more complex in combinatorial spaces. When a search problem is encoded with a discrete representation, "locality" has a meaning solely with respect to a given neighbourhood structure. Hence, although the evolutionary components of these algorithms may operate on a fixed landscape, those neighbourhood structures that are seen by memes, and consequently the set of solutions which are locally optimal, change over time. This impacts both on the design of credit assignment mechanisms, and on the management of the trade-off between global and local search.

The gaps in current understanding are even larger for third generation MAs (3GMAs), which dynamically adapt, and create, memes and their functional definitions, for which purpose many different heuristics and metaheuristics can be used. As a result, depending on the metaheuristic used, at any given time all of the memes available may be present for the first time, and/or some may be duplicated. Therefore, credit assignment mechanisms based on historical records will have limited application, and it is necessary to estimate the value of a meme via some form of sampled measure of its effect in the current population of candidate solutions. Previous results by ourselves and others have shown that a sample size of one (i.e., a single pairing of meme with a candidate solution) creates excessive noise, but clearly testing every meme against every candidate solution may not be viable other than for small sets of memes.

Based on these observations, this paper addresses two outstanding issues concerning credit assignment in general AMAs working in combinatorial search spaces.

1. In the light of recent results on adapting operator choice in EAs, is it better to assign credit to a meme based on an estimate of the extreme benefit, or on the

[1]The terms second and third generation MAs are explained fully in Section 2.

mean benefit it causes? It has been argued that extreme versions, which reward occasional large jumps in fitness rather than small steady improvements, yield better results in EAs, but that is in the context of adapting the global search components, which are responsible for escaping from local optima. In contrast, memes typically represent mechanisms that cause systematic improvements to local optima.

2. Since memes represent local improvement operators, should their value be estimated globally (using information from every pairing of an incident of that meme with a candidate solution) or should it be sampled and estimated locally (separately in different regions of the search space)? Alternatively, for 3GMAs, how should fitness be assigned to a meme which occurs multiple times in a meme population? Different schemes will implicitly create different amounts and types of noise in the sampling process. For example, evidence from a local extreme scheme might fool the meme adaptation mechanism by awarding large credit to an otherwise useless meme that happens to cause a large improvement in a poor candidate solution. Conversely, local mean evidence might lose too much information in the averaging process to drive successful adaptation.

In order to generate insights which can be seen in a context of previous research, we use an existing framework. Taking inspiration from Dawkins' original concept of memes as evolving entities which influence the behaviour of individuals coded for by a population of genes, the coevolutionary memetic algorithms framework (COMA) was designed as a testbed for investigating a range of behaviours and effects in AMAs. Section 3 summarises how starting with simple fixed length pattern-matching memes, and successively building in more complexity, experimental results have shown significant performance benefits over fixed MAs on a range of problems. Using that framework to instantiate both 2GMAs and 3GMAs, we examine all four combinations of global/local and extreme/mean reward, focussing on the sometimes conflicting aspects of effectiveness and efficiency.

The major contribution of this paper is the use of a well-established general framework to generate experimental results that attempt to provide answers to the questions above, and hence provide generic guidance to designers of AMAs. The paper is organised as follows: Sections 2 and 3 review related work and the historical development of COMA to provide context. Sections 4 and 5 describe the algorithmic and experimental setup to support this research. Section 6 reports the results obtained when different reward schemes were tested in exemplars of 2GMAs and 3GMAs. Section 7 discusses the findings in the light of related research, and analyses the changing pattens of meme usage and behaviour in order to explain the results. Finally, Section 8 discusses these results and their significance.

2 Background

The field of memetic computation[2] 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 considering these algorithms to be just local search evolutionary hybrids, Ong et al. (2010) consider memetic computation to be a more general paradigm which uses "the notion of meme(s) as units of information encoded

[2]Often also referred to as memetic computing.

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, for example, in Ting et al. (2010).

In this paper, we restrict ourselves to the broad class of MAs. Introduced by Moscato (1989), these combine population-based global search heuristics (such as EAs) with heuristics that attempt to improve a single solution. Meuth et al. (2009) distinguish between:

- **First Generation MAs.** Defined as "Global search paired with local search."
- **Second Generation MAs.** Defined as "Global search with multiple local optimizers. Memetic information (choice of optimizer) passed to offspring (Lamarckian evolution)."
- **Third Generation MAs.** Defined as "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."

They noted that, at the time of writing, the self-generating MAs and COMA were the only algorithms falling into the 3G class, and went on to propose (but not implement) a fourth generation of MAs in which they suggest: "Mechanisms of recognition, generalization, optimization, and memory are utilized." Arguably, the use of pattern-based memes in COMA fall into this class, and certainly the ADEP framework described in Chen (2010) represents an important step toward such algorithms. Thus, while this paper deals with credit assignment issues which must be dealt with in 2GMAs and 3GMAs, many of the issues will also be present as 4GMAs are developed (it is believed that patents are being processed in this area).

2.1 Credit Assignment in Adaptive Memetic Algorithms

There are several recent examples of the use of multiple meme operators within evolutionary systems. Ong et al. (2006) present an excellent recent review of work in the field of AMAs. This encompasses multi-memetic algorithms (Krasnogor, 1999, 2002; Krasnogor and Smith, 2000, 2001; Krasnogor et al., 2002), the COMA framework (Smith, 2002a, 2003c, 2004, 2007a, 2007b), meta-Lamarkian MAs (Ong and Keane, 2004), hyper-heuristics (Cowling et al., 2001; Burke and Smith, 2000; Kendall et al., 2002; Burke et al., 2003, 2010), and self-generating MAs (Krasnogor, 2004; Krasnogor and Gustafson, 2004).

Essentially, each of these approaches maintains a pool of local search operators available to be used by the algorithm, and at each decision point makes a choice of which to apply. Ong's classification uses terminology developed elsewhere to describe adaptation of operators and parameters in EAs (see Section 2.3). This taxonomy categorises algorithms according to the way that these decisions are made. One way (static) is to use a fixed strategy. Another (adaptive) uses feedback of which operators have provided the best improvement recently, and is further subdivided into external, local (to a deme, or region of search space), and global (to the population), according to the

nature of the knowledge considered. Finally, they note that LS operators may be linked to candidate solutions (self-adaptive). In benchmark studies on five continuous problems, they reported that a global-level adaptation scheme yielded the highest mean best fitness after a fixed number of evaluations compared to the local adaptive scheme, and both outperformed schemes that did not adapt operator probabilities, or did so without reference to search information. One of their main conclusions was that a good deal more research needs to be pursued into the adaptation and credit assignment mechanisms.

A number of subsequent papers have expanded on these themes. Barkat Ullah et al. (2009) proposed an agent-based approach for optimising constrained problems defined over continuous spaces. Here each agent had available to it a suite of local search algorithms, and maintained a local record for each. This was a scalar value in the range $\{-1,1\}$ according to the meme's effect on the feasibility of the candidate solution, and also on the fitness improvement caused, in the last generation.

Caponio et al. (2007) proposed a fast adaptive memetic algorithm that simultaneously adapted the global and local search characteristics according to a measure of (global) fitness diversity. Adapting the global search was done by adjusting the EA's population size and the aggressiveness of mutation to maintain diversity. The probability of applying two very different local search operators was determined using a static external rule, taking as evidence the generation count and the ratio of the current fitness diversity to the most extreme value ever observed. This concept was explored further in Neri (2007a, 2007b). Caponio et al. (2009) extended this idea, using fitness diversity in a probabilistic way (beta distribution) to assign activation probability to the memes.

Nguyen et al. (2009a) proposed static adaption in cellular memetic algorithms. Their approach split the population into groups according to fitness diversity and applied global search to one member from each group, with a blacklist of members that did not benefit from local search. This blacklist method used local historical evidence to bias the global/local search trade-off, and while highly effective, it of course only worked with fixed memes. A more general probabilistic memetic framework was proposed in Nguyen et al. (2009b) to adapt the global/local search trade-off. Using arguments based on the likelihood of generating points within a basin of attraction on a continuous landscape, they proposed to dynamically estimate the probabilities of achieving benefits via local search and global search, and adapting the number of iterations allowed to local search accordingly. This was successfully instantiated using local search traces plus a database of historical points. Notably, this used an extreme measure of improvement to assign credit to memes.

2.2 Credit Assignment in Coevolutionary Systems

If selection is performed separately for the two populations, with memes' fitness assigned as some function of the relative improvement they cause in the solution population, then we have a cooperative coevolutionary system. Bull (1995) conducted a series of more general studies on cooperative coevolution using Kauffman's static NKC model. Bull and Fogarty (1997) examined the evolution of linkage flags in coevolving symbiotic systems and showed that the strategies which emerge depend heavily on the extent to which the two populations affect each other's fitness landscape. In highly interdependent situations, linkage of the two species' chromosomes was preferred—which in our context is equivalent to memes' self-adapting as part of the solutions' genotypes. Bull (1997) also examined the effect of various strategies for pairing members of different populations for evaluation, with inconclusive results. This work was revisited and extended by Wiegand et al. (2001) with very similar findings. Wiegand's

work also considered the number of partners with which a member of either population should be evaluated, which draws attention to the trade-off between accurately estimating the value of an object (solution or meme), and using up evaluations doing so. Parker and Blumenthal (2004) suggested "punctuated anytime learning with samples" as an alternative approach to the pairing problem, using periodic sampling to estimate fitness, but this is more suited to cases where the populations evolve at different rates.

2.3 Credit Assignment in Adaptive Operator Selection

Since the beginnings of the field of evolutionary computation, the question of how to assign the probabilities of applying different operators, and the choice of associated parameters, has been the subject of intense and ongoing scrutiny. A wide range of different strategies has been proposed for adapting operator probabilities in response to their perceived utility (see, e.g., Smith and Fogarty, 1997; Eiben et al., 1999; or a synthesis of the work in Eiben et al. (2007)). There are two principal categories: self-adaptive schemes (where utility is implicitly assumed, via association with fitter solutions that survive selection) and adaptive schemes that track the qualities of offspring produced by different operators and then recalculate probabilities periodically. The use of the intrinsic evolutionary processes to adapt mutation step sizes has long been used in evolution strategies, as described by Schwefel (1981), and evolutionary programming, as described by Fogel (1992). Bäck (1992), and Smith and Fogarty (1996c) applied similar approaches to self-adapt mutation probabilities. The choice of crossover points in recombination operators was self-adapted by Schaffer and Morishima (1987), Smith and Fogarty (1996b), and Smith (2002b). Smith and Fogarty (1996a) combined these to develop more complex generating operators. More recently, Serpell and Smith (2010) have showed that self-adaptation can very effectively govern both the choice and parameterisation of different mutation operators for GAs with permutation representations.

Recent work in the area of adaptive operator selection by, for example, Maturana et al. (2009), and Whitacre et al. (2006) has divided the problem into two areas—first how to assign a quality metric to an operator that changes responsively over time, and second how to allocate probabilities to operators on that evolving basis (see, e.g., papers by Thierens, 2007; Fialho et al., 2008; Maturana et al., 2009; Fialho et al., 2010; Burke et al., 2010). A major proposal emerging from this stream of work is that it may be beneficial to use extreme values, that is, to use the maximum positive difference between offspring and parent fitness, rather than the mean value of the effect of an operator. This is in the spirit of rewarding operators that produce occasional large jumps in fitness rather than those which produce steady, but small, fitness improvements. However, as we have noted above, this relates to the characteristics of global search via the EA.

In a memetic context, clearly it is beneficial to estimate values by evaluating memes in the context of more than one solution, and initial experiments (not shown for reasons of space) show this can be achieved by increasing the selection pressure in the meme population by using tournaments of size 5. Equally clearly, this credit assignment mechanism needs to be responsive to the current (rather than historical) state of the population of candidate solutions. This follows a stream of arguments in evolutionary computation research using the current rather than historical data to model the current population (as in estimation of distribution algorithms) or the underlying search landscape (as in covariance matrix adaptation). Similarly, Thierens (2007) proposed adaptive operator pursuit, and Maturana et al. (2009) trigger forgetfulness of previous operator utility in their dynamic multi-arm bandit approaches. In 3G versions

of COMA, the probability allocation is dealt with by the action of selection in the meme population—as long as the memes' fitness accurately reflects their quality.

Based on the review above, three possibilities can be identified for assigning meme fitness. The first is implicit, that is it uses the fitness of the attached solution. This does not necessarily imply self-adaptation, since the selection processes could be decoupled, but it does imply same-sized populations. Therefore, we do not consider it here as it is less relevant to 2GMAs. The second and third possibilities record the effect of every time a meme is applied, and use either the mean or extreme improvement caused.

3 Previous Findings with the COMA Framework

The COMA framework maintains two parallel populations: one of candidate solutions (genomes) and one of memes, representing local search algorithms to be applied to the genomes. Typically, an EA is applied to both populations, but there is no fixed restriction on the choice of metaheuristic used to adapt either population. In the work to date, the neighbourhood structure used by a meme is defined via pattern matching and substitution, akin to regular expressions. Thus, each meme encodes a condition and action pattern of equal length plus an integer rule_length specifying the number of positions in the pattern string to consider. The memes' rules are shorter than the genomes, so the local search neighbourhood of a meme is defined as the set of solutions obtained by the matching and substitution of genome substrings, and correspondingly the neighbourhood size may vary for each meme-genome pairing. Note that this concept of evolving adaptive neighbourhoods refers to the landscape structure, as opposed to the algorithmic (deme) structures coevolved by Whitacre et al. (2008).

Using full linkage between the two populations, so the memes were self-adapted with the genomes, Smith (2002a) demonstrated successful evolution of rules with the appropriate lengths and symbols to capture structural dependencies for various different types of binary-encoded problems. This was then extended to higher cardinality representations, and successfully applied to the problem of protein structure prediction by Smith (2003c). Using the binary testbed, Smith (2003a) introduced a don't care (#) allele for the rule condition, and examined the use of different pivot rules (i.e., steepest or greedy ascent) and different linkage strategies between the two populations. The results showed that the random pairing of memes and genomes (i.e., uniform credit assignment to memes) was poor, indicating that adaptation rather than merely meme variety is necessary. Assigning credit on the basis of a single meme–genome interaction was found to introduce excessive noise when coupled with a greedy pivot rule, so steepest ascent was more reliable, albeit slower on most problems. In Smith (2007a), an invert allele for actions and a binary flag for the choice of pivot rule were introduced and yielded more robust behaviour across a wide range of test problems.

Smith (2007b) focused solely on the issue of credit assignment in adaptive memetic algorithms using Ong's taxonomy as a guideline. COMA was instantiated with a single step of greedy local search, and a variety of mechanisms for credit assignment in the meme population. The results clearly demonstrated the advantages of a credit assignment mechanism based on more than one meme–genome pairing. A memory-based approach was able to rapidly identify and exploit a problem structure if present, but this faster convergence was disadvantageous on unstructured problems such as MAX-SAT. A local adaptive strategy with a collaboration pool size of two was the most effective and efficient, and the use of mean evidence was better than the extreme evidence case.

These papers represented the development and exploration of the COMA framework, and it should be noted that the most successful credit assignment schemes relied on equal-sized populations. Also, the restriction to a single step of local search simultaneously simplified the management of global/local search, and limited the scalability of the approach to large problems where local search is more effective (e.g., OneMax). In Smith (2010), a flag was added to the memes denoting a binary choice of whether local search should run for one iteration, or until it reached local optimality. Early results showed that the choice of fixed strategy was problem dependent, but the adaptive schemes were always competitive. An investigation of different memepool sizes (which affects the size of sample in the credit assignment) showed that provided adequate selection pressure was present, a large number of memes aided the overall search process.

4 A Framework for Self-Adaption and Coevolution of Memes and Genes

The pseudocode in Algorithm 1 illustrates the COMA framework. Note that although this pseudocode assumes synchronous evolution, this need not in general be the case. For the sake of clarity, we have omitted some of the parameters, for example, $Recombine(ParentSet, parent1id, parent2id)$ is assumed to return a copy of the first parent with probability $1 - Px$ (where Px is the probability of applying crossover).

The representation of the memes is a tuple *<Pivot, Depth, Pairing, Move>*. The representation of the tuple elements leads naturally to the choice of evolutionary variation operators. The *Pivot* element is naturally binary. The *Depth* element is mapped as an integer representing the computational effort allowed (maximum number of calls to fitness function). An arbitrarily large number is used to signify that search should always progress until a local optima is reached. Algorithm 2 illustrates the process of applying a meme to a solution. Note that this paper uses Lamarkian learning and the pattern-based specification of meme neighbourhoods described below.

The *Pairing* element is one of {*Self-Adaptive, Random, Fitness_Based*} and determines how memes are created and applied to solutions. As is illustrated in Algorithm 2, a range of behaviours from self-adaptive through collaborative coevolution to random meme drift can be obtained by fixing the elements; and selectively allowing mutation to operate on them creates various different adaptive schemes. Note that a wide range of selection operators can be applied to either (or both) populations since they work on the basis of fitness, and are independent of the representation.

This framework is designed to be generic in the way that move operators are described; for example, they could be GP-like expressions as per Fukunaga (2008). Here they are encoded as condition:action pairs, which specify one pattern to be looked for in a genome, and another to replace it. Although this representation at first appears to be very simple, it has the potential of representing highly complex moves via the use of symbols to denote not only single/multiple wildcard characters (in a manner similar to that used for regular expressions in Unix) but also the specifications of repetitions and iterations. Further, permitting the use of different length patterns in the condition and action parts of the rule gives scope for cut and splice operators working on variable length solutions. The neighbourhood of a genome i then consists of i itself, plus all those points where the substring denoted by condition appears in the representation of i and is replaced by the action. To give an example, a rule 1#0 $\rightarrow$ 111 matches the binary string 1100111000 in positions 1, 2, 6, and 7, so the neighbourhood is the set {**110**0111000, **111**0111000, 1**111**111000, 1100**111**00, 110011**111**0}. The string is not treated as toroidal,

Algorithm 1 Pseudocode definition of COMA algorithm

```
COevolving Memetic Algorithm for Binary Coded Problems :
Begin
  /* Given populations P of μs solutions and M of μm memes */
  initialise P and M randomly ;
  set generations = 0;
  set evaluations = 0;
  Repeat Until (run_termination condition is satisfied)
  Do

    /* Create μs solution offspring and store parent ids */
    For i := 1 To i = μs Do
      /* Select pool of parents and store ids */
      set FirstParent[i] = Select_One_Genome_Parent(P);
      set SecondParent[i] = Select_One_Genome_Parent(P);
      /*  Create offspring by recombination and mutation */
      set Offspring[i] = Recombine(P, FirstParent[i],SecondParent[i]);
      Mutate(Offspring[i]);
      set i = i +1;
    Od

    /* Create μm meme offspring according to pairing */
    For i := 1 To i = μm Do
      /* Select parents of the new meme and store their indices */
      set Pairing = Get_Pairing(M,i);
      If (Pairing = SelfAdaptive) Then
        set MemeParent1[i] = FirstParent[i];
        set MemeParent2[i] = SecondParent[i];
        /* note this requires μm = μs.  */
      Fi
      Else If (Pairing = Fitness_Based) Then
        set MemeParent1[i] = Select_One_Meme_Parent(M);
        set MemeParent2[i] = Select_One_Meme_Parent(M);
        /* Note selection mechanism may not be same in each population */
      Fi
      Else
        set MemeParent1[i] = RandInt(1,μm);
        set MemeParent2[i] = RandInt(1,μm);
      Esle
      /* Create meme offspring via recombine parents and mutation */
      set NewMemes[i] = Recombine(M, MemeParent1[i],MemeParent2[i]);
      Mutate(NewMemes[i]);
      set i = i+1;
    Od

    /* Apply local search to Offspring Using Memes */
    For i := 1 To i = μs Do
      If (Pairing = SelfAdaptive) Then
        set meme = i;
      Fi
      Else
        set meme = Select_Random(NewMemes);
      Esle
      set evals_used = Pair_Meme_With_Solution(Offspring[i],NewMemes[m]);
      /* the procedure also updates the meme and genome fitness */
    Od
    set P = Offspring;
    set M = NewMemes;
  Od
End
```

Algorithm 2 Pseudocode for application of meme to candidate solution

```
Pair_Meme_With_Solution(meme m , solution s):
Begin
  set maxdepth = Get_Depth(m);
  set pivot = Get_Pivot(m);
  set evals_used = 0;
  set original_fitness = Get_Fitness(s);
  set incumbent = s;
  set Locally_Optimal False;
  While ((evals_used < maxdepth) and (Locally_Optimal = False))
  Do
    /* Apply the meme's pattern matching rule */
    /*  to create a ordered set of neighbours */
    set Neighbours = Apply_Rule_To_Offspring(incumbent,m);
    Evaluate_Fitness(Neighbours);
    /* Pivot rule of meme determines choice of neighbour */
    If (pivot = Greedy) Then
      set best = Get_First_Better_Than(incumbent, Neighbours);
      set evals_used = evals_used + PositionInSequence(best, Neighbours);
    Fi
    Else
      set best = Get_Best(Neighbours);
      set evals_used = evals_used + | Neighbours|;
    Esle
    If (Get_Fitness(best) < Get_Fitness(incumbent)) Then
      set Locally_Optimal = True;
    Fi
    Else
      set incumbent = best;
    Esle
  Od
  set Δfitness = Get_Fitness(incumbent) - original_fitness;
  Update_Solution_Fitness(s, Get_Fitness(s);
  If (Lamarkian_Learning) Then
    set s = incumbent;
  Fi
  Update_Meme_Fitness(m, Δfitness, evals_used);
  return evals_used;
End
```

and the neighbours are evaluated in a random order so as not to introduce positional bias into the local search when greedy ascent is used.

5 Test Suite and Methodology

A range of well understood test problems was used to examine the performance of various 2GMAs and 3GMAs using combinations of global/local and extreme/mean improvements as evidence to the meme update mechanisms. A range of binary-coded combinatorial problems was used; some of these are standard testbed functions for EAs, while others were specifically designed to probe certain behaviours.

For this paper, we focussed on three measures of performance. The effectiveness of the search algorithm is measured by the success rate (SR)—the number of runs finding the global optimum. The efficiency is measured by the average evaluations to solution (AES). The reliability is measured by the mean best fitness observed per run (MBF). For each problem, 100 runs were made (10 each for instance for the MAX-SAT problems), each continuing until the global optimum was reached, subject to a maximum of 500,000 evaluations. The reason for the large cutoff value was to try to avoid skewing results, as

can happen with an arbitrarily chosen lower cutoff, rather than to be indicative of the amount of time available for a real world problem. Note that since one iteration of a local search may involve several evaluations, algorithms are compared strictly on the basis of the number of calls to the evaluation function. To analyse the experimental results, we used appropriate nonparametric tests (Mann-Whitney for two groups, Kruskal-Wallis for three or more, with post hoc pairwise comparisons) to identify whether there were significant differences between groups in the SR, AES, and MBF. In every case where a difference is reported, it should be read as being statistically significant with over 95% confidence.

5.1 Test Problems

The first set of problems used is composed of 16 subproblems of Deb's 4-bit fully deceptive function described in Bäck et al. (1997). The fitness of each subproblem i is given by its unitation $u(i)$, that is, the number of bits set to one:

$$f(i) = \begin{cases} 0.6 - 0.2 \cdot u(i) : & u(i) < 4 \\ 1 : & u(i) = 4 \end{cases} \tag{1}$$

Versions of trap were used with lengths taken from the set {40, 80, 120, 160, 200, 400, 800, 1000, 2000}.

The royal road function used is a simple R1 type with fitness rewards for groups of eight contiguous genes all set to 1. The set of lengths used was {64, 96, 128, 160, 212, 256, 512, 1024}.

Watson's highly epistatic HIFF function rewards matching pairs of adjacent bits in a solution s, that is,

$$f_1 s = \sum_{i=0}^{l/2-1} 1 - XOR(s_{2i}, s_{2i+1}) \tag{2}$$

and this process is applied recursively, so that a problem of size $l = 2^k$ has k levels. In each ascending level, the number of blocks is reduced by a factor of two, and the fitness awarded for each matching pair is increased by a constant factor, in our case, two. This problem has a number of Hamming suboptima, and two global optima corresponding to the $u(i) \in \{0, 1\}$. Problem sizes $l \in \{32, \ldots, 512, 1024\}$ were used, corresponding to 3–10 levels. Note that for l >16, the length of the blocks to be identified at the highest levels far exceeded the maximum rule length.

The MAX-SAT problem is a classical combinatorial optimisation problem, consisting of a number of Boolean variables and a set of clauses built from those variables. A full description and many examples can be found on the SATLIB (2010) website. For lengths of 50 and 100 variables, the first 10 were taken from the sets of uniformly randomly created satisfiable instances around the phase transition (in terms of hardness) where there are approximately 4.3 clauses per variable.

In the light of previous results, and given that Maturana et al. (2009) have asserted that "it is well-known that EAs do not perform well on MAX-SAT without using specialized operators," a simplified version of the SAW algorithm (Eiben and van Hemert, 1999) was implemented for the MAX-SAT problems. Each constraint/clause is initially given a weight of +1, and the fitness of a given solution is the sum of the weights of the constraints it satisfies, normalised by the current sum of all weights. In every generation, the current best solution is analysed and a value of +1 is added to the weight of each constraint which is not satisfied by that solution, which thus focusses the search toward those clauses.

5.2 Algorithms

In previous papers, we have repeatedly shown that a vanilla GA is always outperformed by 3GMAs, and that a simple 1GMA (SMA) using a bit-flipping hill-climber is always outperformed except on certain MAX-SAT problems (e.g., Smith, 2007a, 2010). With the addition of the SAW mechanism to MAX-SAT, the simple MA is now outperformed on those problems. Therefore, especially in view of the fact that the success rates of the GA and SMA are almost always zero on the longer problems used here, the results for these algorithms are omitted for the sake of clarity.

For the population of candidate solutions, a generational genetic algorithm was used, with no elitism, and deterministic binary tournaments to implement $Select\_One\_Genome\_Parent()$. The population size μ_s was 400. One point crossover was applied with probability 0.7, followed by self-adaptive mutation using the scheme outlined in Smith (2001, 2003b), and Stone and Smith (2002). Rather than attempting to adapt a continuous mutation rate parameter, each solution encodes a choice from a discrete set of mutation rates, and this mutation choice gene is itself subject to mutation with a probability 0.01. These choices were taken as standard, and no attempt was made to tune them to individual problems.

The 2GMA was instantiated using five static memes and the adaptive operator pursuit mechanism described in Thierens (2007). In light of the problems chosen above, five greedy search operators were implemented. Using the notation above, with I denoting invert, these were $\{\# \rightarrow I\}$ with depth one, or to local optimality, $\{\#\# \rightarrow II\}$, $\{\#\#\#\# \rightarrow IIII\}$, and $\{\#\#\#\#\#\#\#\# \rightarrow IIIIIIII\}$, all with depth one.

$P(m)$, the probability of selecting meme m to be used in a genome-meme pairing, is initialised to 0.2. The improvement observed in the pairing is recorded. At the end of each generation of the GA, the best performing meme m^* is noted, and for each meme m, $P(m)$ is updated according to:

$$P'(m) = \begin{cases} P(m) + \beta(P_{\max} - P(m)): & m = m^* \\ P(m) + \beta(P_{\min} - P(m)): & \text{otherwise} \end{cases} \tag{3}$$

using the suggested values $P_{\min} = 0.1$, $P_{\max} = 0.6$, and $\beta = 0.8$. To keep the memes static, the probability of crossover or mutation was set to zero in the meme population.

To instantiate the third generational MA, $Select\_One\_Meme\_Parent()$ used deterministic tournaments of size 5. No crossover was used in the meme population, so memes were produced by copying selected parents and then applying mutation. The pivot rule in all memes was fixed to be greedy. The mutation rate in the meme population was set to be self-adaptive, as for the solution population. To this end, a single gene in each meme genome P_{mm} encoded for one of the values $1.0/RL \times \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 1.0, 2, MIN(0.25 \times RL, 5.0), MIN(0.25 \times RL, 10.0)\}$ where RL is the maximum allowed rule length (16). The global meme strategy adaptation rate P_{mg} takes a value of 0.1.

During meme mutation, with probability P_{mg}, an $\mathcal{N}(0, 2)$ normal deviate is added to the encoded rule length, which is then truncated to the range [0, 16]. An identical process adapts the encoded meme depth with step size 1,000 and range [0, 25,000]. Next, again with probability P_{mg}, the value of P_{mm} is randomly reset. Finally, the condition and action parts of the rules are subjected to allelewise mutation with probability P'_{mm}.

For both 2GMAs and 3GMAs, the meme depth is taken to represent the total number of solutions evaluated during local search, that is, taking into account the size of the neighbourhood searched. The search was terminated when no improvement was found, so that the depth represents a maximum value available rather than the absolute

Table 1: Explanation of symbols used to describe algorithms in this paper. Note that the term memotype is used analogously to genotype as a specific set of allele values that may have several copies in a meme population. Note also that the sets of fitness improvement include all the steps taken during a local search.

Symbol	Algorithm described
P_{mm}	Locuswise probability of mutating condition and action parts of meme rules.
P_{mg}	Componentwise probability of applying mutation to other elements of a meme.
RL	Maximum length of rule in memes.
Xl	Extreme-local strategy. Uses maximum observed fitness improvement when applying an individual meme, that is, sampled locally in solution space.
Xg	Extreme-global strategy. Uses maximum observed fitness improvement from all applications of a memotype, that is, sampled globally across solution space.
Ml	Mean-local strategy. Uses average observed fitness improvement when applying an individual meme, that is, sampled locally in solution space.
Mg	Mean-global strategy. Uses mean observed fitness improvement from all applications of a memotype, that is, sampled globally across solution space.

value used. However, each neighbourhood is sampled fully, even if doing this slightly exceeds the depth value. Hence, a depth of 1 means that one iteration of local search is applied, but a depth of 1,000 could mean between 1 and 1,000 iterations, depending on the length of the solution representation, the meme's rule length, and the specificity of its condition.

As well as the comparing memes according to the extreme (X) or mean (M) improvement they caused, a no-duplicates strategy was used to evaluate whether it was possible to link meme actions to particular regions of solution space. In a local strategy (Ml or Xl), each meme maintained its own fitness record, so different copies of the same meme can have different fitnesses. Alternatively, in a global strategy (Mg or Xg), although multiple copies of memes were allowed, a single record was maintained for each meme genotype. To aid the reader, Table 1 specifies the meaning of the various symbols used.

6 Comparative Results

6.1 Results for Second Generational Memetic Algorithms

Table 2 summarises the results obtained with the 2GMA using adaptive operator pursuit based on either extreme or mean improvement to allocate the meme probabilities. The lengths of the problem that were successfully tackled were less than the 3GMA versions. As can be seen, in most cases, the extreme version is less effective (lower SR) slower (larger AES), and less reliable (lower MBF). These results are confirmed by the appropriate nonparametric tests with over 95% confidence.

6.2 Performance of Reward Schemes in 3G MAs

For those problem instances where at least one, but not all, algorithms located the global optimum, Table 3 summarises the success rates achieved, the mean time taken to locate the global optimum, and the mean best fitness observed. The mean best fitness achieved and the number of evaluations at which this was done are also shown graphically in Figure 1. Given the effectiveness of these algorithms, we do not present results from short problem lengths where all algorithms performed equally well, but rather for each problem focus on those challenging instance lengths where different patterns of

Table 2: Comparison of search effectiveness (success rate), efficiency (AES), and reliability (MBF) for different reward strategies with a 2GMA with five memes. For each problem and metric, the best results are shown in bold.

Function	length	SR		AES (1,000s)		MBF	
		M	X	M	X	M	X
HIFF	32	**100**	**100**	**26.29**	27.10	**100**	**100**
	64	**100**	**100**	**167.79**	223.36	**100**	**100**
	128	40	**43**	**550.76**	801.26	**90.20**	89.34
	256	0	0	0.00	0.00	66.00	53.45
Royal road	64	**100**	**100**	**282.30**	284.86	**100**	**100**
	96	97	99	532.31	588.73	99.73	**99.91**
	128	**60**	40	**772.76**	899.24	**96.42**	95.2
	160	**5**	1	**906.62**	998.44	**88.20**	84.85
	212	0	0	0.00	0.00	**72.64**	67.76
Trap	40	**100**	**100**	**87.47**	120.71	**100**	**100**
	80	**100**	**100**	**339.26**	532.71	**100**	**100**
	120	**91**	24	**671.70**	897.57	**99.76**	98.14
	160	**13**	0	844.13	0.00	**97.47**	91.76
	200	0	0	0.00	0.00	**94.17**	86.16
MAX-SAT	50	43	**69**	**3196.50**	3572.36	99.53	**99.80**
	100	**18**	15	2960.76	**1482.64**	**99.1**	**99.1**

Table 3: Comparison of search effectiveness (success rate), efficiency (AES), and reliability (MBF) for different reward strategies on problems. Instances solved by all or no algorithms omitted. 3GMAs with 400 memes, elitism in genome population, depth of local search measured in evaluations. For each problem, the best values of the algorithm are shown in bold for the three metrics.

Problem	length	Success rate				AES (1,000s)				MBF			
		Mg	Ml	Xg	Xl	Mg	Ml	Xg	Xl	Mg	Ml	Xg	Xl
HIFF	128	**100**	**100**	**100**	**100**	**88.44**	88.59	121.31	117.09	**100**	**100**	**100**	**100**
	256	**99**	**99**	95	**99**	**379.65**	403.02	585.15	536.54	**99.89**	99.83	99.1	99.83
	512	**31**	**31**	8	13	607.74	672.40	**242.14**	492.66	83.44	**84.07**	73.03	77.09
	1024	4	**6**	2	1	702.07	**549.29**	690.30	562.32	66.07	**67.27**	61.76	62.30
Royal road	64	97	**100**	98	**100**	145.22	**120.33**	145.33	121.90	99.61	**100**	99.74	**100**
	96	91	94	94	**96**	337.60	**295.10**	353.99	302.31	99.19	99.46	**99.64**	99.43
	128	73	79	73	**84**	619.99	518.61	598.19	**515.57**	97.87	98.35	97.51	**98.88**
	160	39	**56**	29	52	878.09	**763.80**	878.85	795.60	95.85	**97.15**	94.35	97.00
Trap	120	**100**	**100**	97	**100**	**90.73**	91.07	362.55	227.24	**100**	**100**	99.96	**100**
	160	**100**	**100**	82	96	148.01	**130.37**	362.40	317.69	**100**	**100**	99.65	99.92
	200	**100**	**100**	73	90	**185.28**	195.63	408.65	409.56	**100**	**100**	99.35	99.75
	400	**98**	97	37	58	392.71	393.42	**342.87**	415.64	**99.97**	99.94	97.33	98.55
	600	**84**	81	39	46	**579.17**	580.22	583.40	586.26	99.81	**99.84**	96.46	97.89
	800	**56**	49	26	35	649.93	**593.11**	614.61	688.16	**99.41**	99.21	94.98	96.34
	1000	33	**45**	21	24	721.72	697.17	672.05	**631.44**	**99.48**	98.58	94.46	94.30
	2000	11	10	**13**	12	**563.74**	618.81	800.12	720.87	90.07	89.84	**91.25**	91.06
MAX-SAT	50	74	78	**96**	94	1958.24	1740.9	**915.4**	1134.2	99.8	99.85	99.98	**99.90**
	100	28	30	**57**	52	4334.5	**4001.8**	6123.9	4521.7	99.63	99.64	**99.87**	99.85

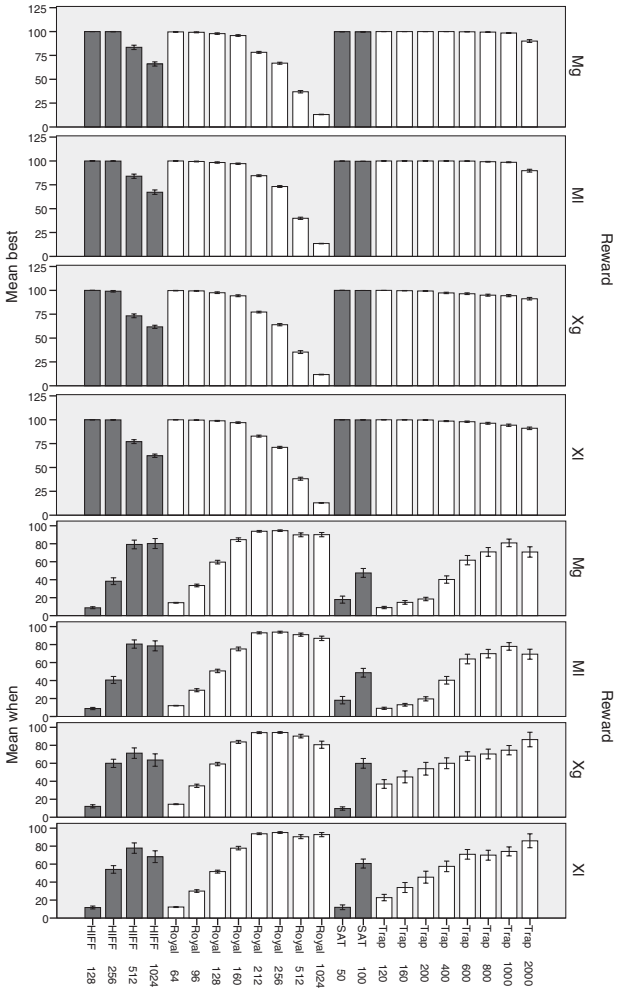

Figure 1: Comparison of mean best fitness (top) and when observed (divided by 10,000; bottom) for different reward functions. Bars represent mean values with error bars at 95% confidence intervals.

behaviour begin to emerge. As is discussed later, every instance of the OneMax function was solved in the first generation by all algorithms.

In order to examine the sensitivity of the results to different parameter settings, further experiments were run with different population sizes, different selection methods (including with elitism not present in the genome population), with various fixed mutation rates for the genome and condition/action parts of memes, and with different meme strategy adaptation rates. In every case, the patterns of results matched those presented here, so for the sake of clarity, those results are omitted but will be available with the algorithm code from the author's website.[3]

Pooling the results and testing for significant differences between results yields the following.

[3]http://www.bit.uwe.ac.uk/~j4-smith/COMA

- All schemes are equally effective at locating global optima: there is no difference between the success rates for different reward schemes.
- The reward schemes based on mean fitness find higher quality solutions than those based on extreme fitness.
- The reward schemes based on mean fitness find their best solutions faster than those based on extreme fitness.
- The reward schemes based on local fitness find higher quality solutions than those based on global fitness, and the time taken is not significantly different.
- Looking at the combination of reward type and locality, the best solutions found by Ml are of significantly higher quality, and are found significantly faster, than those found by Xl or Xg.
- Mg lies between Ml and the two global schemes for quality and speed. The differences between the two mean schemes are not significant. The differences are significant for speed but not quality between Mg and Xl and are significant for both between Mg and Xg.
- Xl finds significantly higher quality solutions than Xg, and the time taken is not significantly different.

Analysing the results by individually functions, we start by noting that for the OneMax function there are no differences in performance. With a population of 400 memes randomly initialised, there is an above 93% chance that the initial population will contain a meme representing a simple bit-flipping hill-climber. For the sake of fairness, the same sets of 100 seeds were used for all comparison runs, and a brief inspection of the results confirmed that in every run the global optimum was located within the first generation, regardless of problem length.

For the MAX-SAT problems, there was no difference in either the quality of the best solution identified, or in the number of evaluations taken to identify it. This is despite the apparently higher success rates for extreme-based rewards.

For the trap functions, Ml and Mg do not differ significantly for either mean best fitness or speed, but on either measure, both are significantly better than Xl or Xg. Xl locates better solutions than Xg, and does so faster, but not significantly faster.

For the royal road functions, the only significant difference in the quality of the solutions is that Ml finds better solutions than Xg. In terms of speed, the only significant difference is that Ml is faster than Mg.

For the HIFF functions, there is no difference in the time to locate the best solutions found, but there is a difference in the quality. Both Ml and Mg find better solutions than Xg, and with 93% confidence, Ml finds better solutions than Xl.

7 Discussion and Analysis

Having noted that both 2GMAs and 3GMAs gave better results with mean-based rather than extreme-based reward schemes, we start by contrasting this with results obtained elsewhere, then look for evidence of the suggested reasons in the patterns of evolving meme usage (2GMA and 3GMA) and behaviour (3GMA).

7.1 Analysis of Relationship to Other Algorithms and Results

Although many of the 2GMA schemes described in Section 2.1 use extreme-based credit, there is little analysis of the effects of this as a design choice. Within adaptive operator selection in EAs, proponents of extreme-value based credit assignment have demonstrated it with various operator adaptation policies. For example, Fialho et al. (2008) show results on OneMax that track the optimal choice of mutation operator, although they do not compare their results against mean-value schemes. Maturana et al. (2009) compare mean-based and extreme-based policies on MAX-SAT problems, but using the more complex COMPASS assignment scheme, which also takes into account the diversity of the offspring. Perhaps more pertinently, both of these papers focus their attention on very low population sizes: $(\mu + \lambda)$ with $\mu = 1$ and 3, respectively. Later papers such as Fialho et al. (2010) performed a deeper analysis of extreme and average-based credit assignment in conjunction with multi-armed bandits and adaptive pursuit (Thierens, 2007) as methods for operator selection. Simulated results on artificial problems showed that the extreme method was either risk-taking, or risk adverse, depending on the operator selection scheme, whereas the mean reward based scheme was unbiased in either setting. Tellingly, on a royal road function with a (100, 100) GA, no statistically significant difference could be observed in between the behaviour of the two approaches to credit assignment, or to a simultaneous scheme. Our hypothesis is that when their scenarios show a difference, this is often because of the focus on tiny populations, where a major factor is a loss of diversity and the possible need to escape from either a local optimum or a plateau, hence the preference for risk-taking methods.

In contrast, our observations show that on both 2GMAs and 3GMAs there is strong evidence against the use of extreme value based schemes. Rather than casting doubt on the results for adaptive operator selection, this points to a fundamentally different role played by memes as opposed to global search operators. Whereas the latter play a vital role in maintaining diversity with the genome population, memes (at least in the context of a memetic algorithm) play a role in intensifying search toward local optima. In fact, in 2GMAs or 3GMAs, there is even a drive toward the subset of points which are locally optimal for a number of different operators. As a consequence, most of the improvement steps arising from a genome-meme pairing will be relatively small.

Thus, whereas large individual improvements arising from a chance mutation event may signify escape from a local optimum, in the memetic context, it is far more likely to signify that the genome involved in the pairing had a low initial fitness. Assigning a high credit to that meme, and therefore increasing its usage in subsequent generations, is unlikely to yield fitnesses in the genome population if the genome is only present in low numbers (because of its relatively low fitness).

To carry this thought experiment further, it is not hard to imagine a meme that enables a genome population to make the transition from a relatively low fitness region (local optima, or plateau) to a higher region, but which does not consistently improve genomes belonging to higher regions, or may even be deleterious to them on average. This situation is analogous to well-known arguments about the diminishing optimal mutation rates for OneMax. In this case, an extreme scheme will still give a high reward to that meme as long as there are a few genomes present from the original regions, even if other memes would be better used.

To examine whether there is evidence for this hypothesis, Figure 2 plots the maximum genome fitness and the relative probabilities of applying the five meme types against evolutionary time in the 2GMA in typical runs for three problems. Looking at the middle row (royal road) at around generation six, one meme is clearly associated with

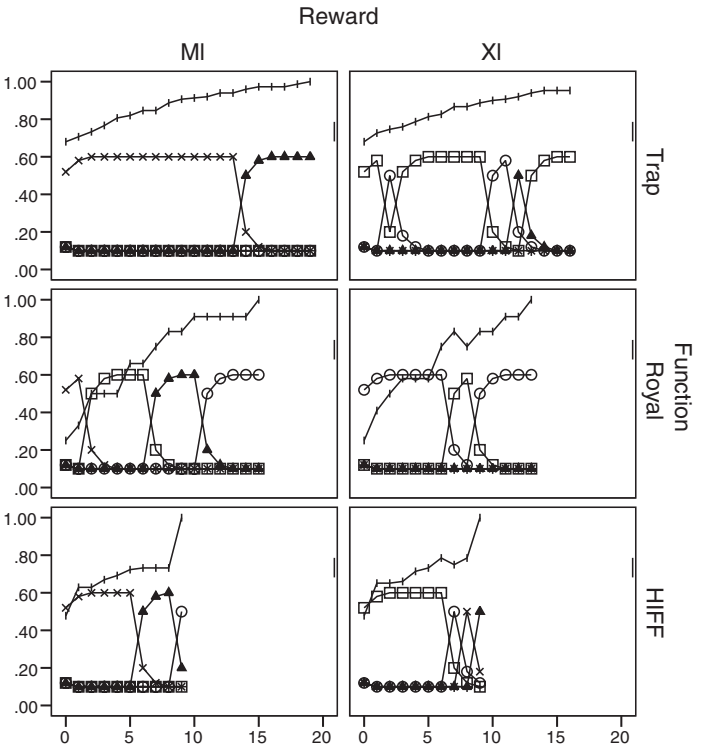

Figure 2: Evolution of meme behaviour for 2GMA separated by fitness function (rows) and reward (column). Each scatter plot has generations for the x axis; y axes are logarithmic.

an increase in the maximum fitness, but as that value plateaus around 10 generations, the meme usage drops off rapidly, to be replaced by a more suitable meme, which causes further fitness increases. Looking at the patterns for the Xl in the same time frame, there is a brief increase in the usage of a meme which is presumably associated with such a low fitness solution, since the maximum fitness actually decreases in that period. A similar pattern can be seen around generation seven in the bottom row (HIFF function).

Figure 3 plots the evolution of the maximum genome fitness, maximum and mean meme fitness, and mean number of evaluations used per local search for typical runs of the 3GMA. Again, exactly the same phenomenon can be seen for the royal road, where following a transition, the Xl scheme shows a prolonged increase in maximum meme fitness, at a period where the genome fitness is stagnating or even falling. In contrast, the peak in meme fitnesses associated with the Ml algorithm is far shorter, and the genome fitness continues to climb.

7.2 Analysis of Evolving Meme Behaviour

Figure 3 shows examples of the fitness patterns and the evolution of the global-local search trade-off (controlled by the depth part of the memes) on the different functions with Ml and Xl rewards. A single representative run is shown, as although similar patterns of behaviour were observed in all the runs inspected, variability in when those occurred tends to obscure the patterns when averages from several runs were plotted.

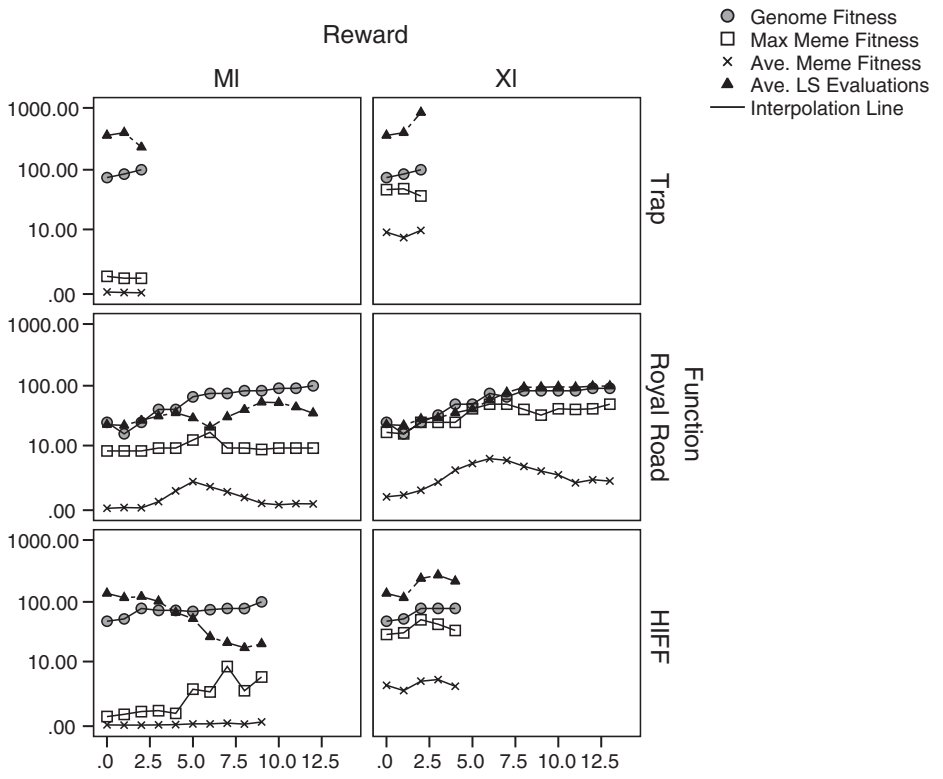

Figure 3: Evolution of different aspects of global-local search balance separated by fitness function (rows) and reward (column). Problem lengths are 1,000 (trap), 256 (royal road), and 512 (HIFF). Each scatter plot has generations for the x axis; y axes are logarithmic.

Figure 4 shows the different patterns of local search behaviour exhibited by the memes in the same runs. To aid viewing in the same scale on the y axis, the mean meme lengths are multiplied by 10 and the mean numbers of evaluations used per local search are divided by 10.

From Figure 4, the following patterns are apparent:

- For the trap function, the performance of both the algorithms uses a large proportion of local search: the 1,000 bit problem is solved within three generations of the GA. Within that time, memes in the Xl population rapidly evolve to length four, and the conditions become less specific than the 67% starting ratio.

- For the royal road functions, the pattern is quite different. In the Ml population, we can see the rise and fall of a meme with high unitation, low specificity, and length 8, that is, a magic bullet of ######## $\rightarrow$ 11111111. However, this meme will of course match in almost every part of the genome, but only definitely bring improvement when aligned with block borders. As noted above, this meme then disappears—the mean meme length drops and the ratio of local search to global search stays relatively low. In other words, given the ability of the EA crossover operator to bring together coadapted blocks of genes, the meme population adapts

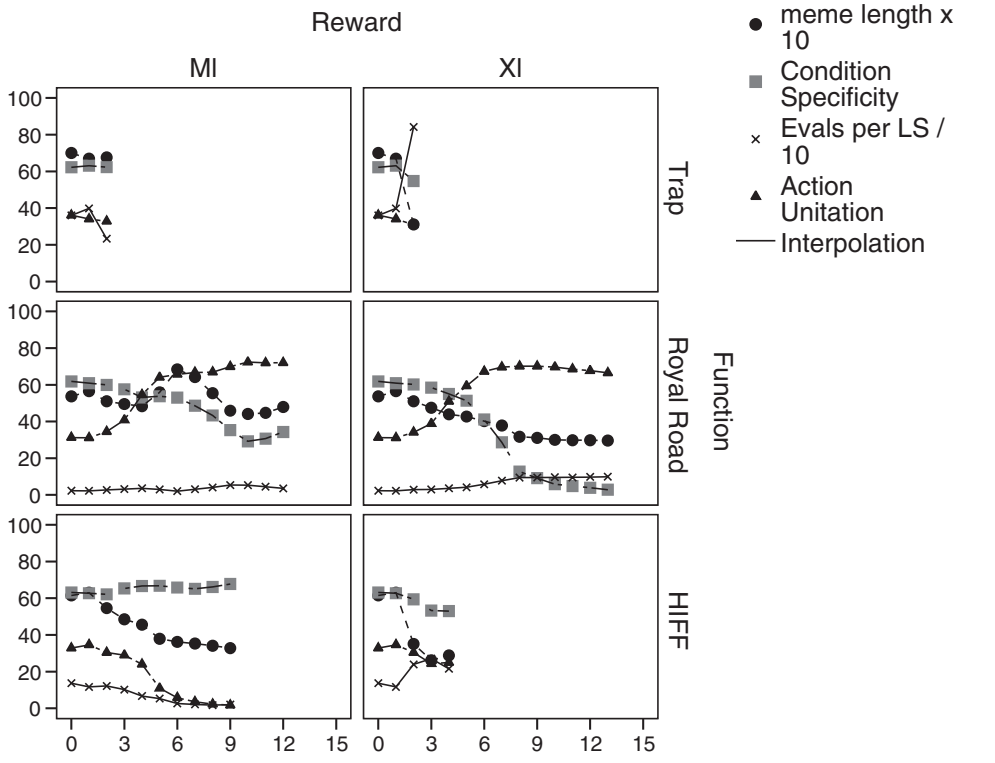

Figure 4: Evolution of meme behaviour separated by fitness function (rows) and reward (column). Problem lengths are 1,000 (trap), 256 (royal road), and 512 (HIFF). Each scatter plot has generations for the x axis; y axes are logarithmic.

to take on a role more akin to a systematic repair mechanism. By contrast, in the Xl meme population, the killer meme is never discovered; rather, short nonspecific memes take over and consume more of the evaluation budget, before uncovering the global optimum.

- On the HIFF function, the Ml algorithm clearly settles on short, nonspecific, low unitation memes which favour the all-zeroes solution within around six generations. The Xl population maintains a unitation of around 30% and consumes more evaluations per local search before improvements are found.

It must be stressed that these are results from single runs—albeit randomly selected. However, the different patterns of behaviour exhibited by the memes evolved under the two reward schemes support the hypothesis that selecting memes on the basis of the mean improvement they cause means they can be more closely coupled to the current genome population. This enables a greater synergy between the global search provided by the evolutionary algorithm and the local improvement caused by the memes.

8 Conclusions

This paper set out to answer two important questions for 2GMAs and 3GMAs: is basing the reward on the extreme improvement observed from a meme preferable to the mean

improvement? And, should the reward be estimated via global, as opposed to local, sampling?

The results strongly support a negative answer to both questions, in contrast to much accepted practice. We propose that this arises from the very different roles of the genetic operators in global (evolutionary) search and local improvement operators represented by memes, and that using extreme rewards, especially when amortised over the global search space, can cause the two aspects of search to become rather decoupled. A close examination of the evolving patterns of meme usage in a 2GMAs, and of meme behaviour in a 3GMA, supports this view, and permits a positive answer to a third question: for 3GMAs, do simple coevolutionary models provide enough information to adapt the local-global search trade-off via the encoded depth of local search?

The use of the COMA framework to instantiate two very different types of AMA with very different methods for allocating the probabilities of using memes, but yielding the same patterns of results, supports our belief that these results have generic importance for the design of AMAs in general.

References

Bäck, T. (1992). Self adaptation in genetic algorithms. In F. Varela and P. Bourgine (Eds.), *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pp. 263–271.

Bäck, T., Fogel, D., and Michalewicz, Z. (Eds.). (1997). *Handbook of evolutionary computation*. Bristol, UK: Institute of Physics Publishing.

Barkat Ullah, A. S. S. M., Sarker, R., Cornforth, D., and Lokan, C. (2009). AMA: A new approach for solving constrained real-valued optimization problems. *Soft Computing*, 13(8–9):741–762.

Bull, L. (1995). Artificial symbiology. PhD thesis, University of the West of England.

Bull, L. (1997). Evolutionary computing in multi agent environments: Partners. In T. Bäck (Ed.), *Proceedings of the 7th International Conference on Genetic Algorithms*, pp. 370–377.

Bull, L., and Fogarty, T. (1997). Horizontal gene transfer in endosymbiosis. In *Proceedings of the 5th International Workshop on Artificial Life : Synthesis and Simulation of Living Systems (ALIFE-96)*, pp. 77–84.

Burke, E., and Smith, A. (2000). Hybrid evolutionary techniques for the maintenance scheduling problem. *IEEE Transactions on Power Systems*, 15(1):122–128

Burke, E., Kendall, G., and Soubeiga, E. (2003). A TABU search hyperheuristic for timetabling and rostering. *Journal of Heuristics*, 9(6):451–470.

Burke, E. K., Curtois, T., Hyde, M. R., Kendall, G., Ochoa, G., Petrovic, S., Rodríguez, J. A. V., and Gendreau, M. (2010). Iterated local search vs. hyper-heuristics: Towards general-purpose search algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8.

Caponio, A., Cascella, G., Neri, F., Salvatore, N., and Sumner, M. (2007). A fast adaptive memetic algorithm for online and offline control design of PMSM drives. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):28–41.

Caponio, A., Neri, F., and Tirronen, V. (2009). Super-fit control adaptation in memetic differential evolution frameworks. *Soft Computing*, 13(8–9):811–831.

CEC-2003. (2003). *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003).* Piscataway, NJ: IEEE Press.

Chen, X. (2010). An algorithm development environment for problem-solving. In *Proceedings of the 2010 International Conference on Computational Problem-Solving (ICCP)*, pp. 85–90.

Chen, X. S., Ong, Y. S., Lim, M. H., and Tan, K. C. (2011). A multi-facet survey on memetic computation. *IEEE Transactions on Evolutionary Computation,* 15(5):591–607.

Cowling, P., Kendall, G., and Soubeiga, E. (2001). A hyperheuristic approach to scheduling a sales summit. In *Practice and theory of automated timetabling. Lecture Notes in Computer Science,* Vol. 2079 (pp. 176–190). Berlin: Springer.

Eiben, A., and van Hemert, J. (1999). SAW-ing EAs: Adapting the fitness function for solving constrained problems. In D. Corne, M. Dorigo, and F. Glover (Eds.), New ideas in optimization (Chap. 26, pp. 389–402). New York: McGraw-Hill.

Eiben, A., Hinterding, R., and Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation,* 3(2):124–141.

Eiben, A., Michalewicz, Z., Schoenauer, M., and Smith, J. (2007). Parameter control in evolutionary algorithms. In F. G. Lobo, C. F. Lima, and Z. Michalewicz (Eds.), *Parameter setting in evolutionary algorithms, studies in computational intelligence,* Vol. 54, (pp. 19–46). Berlin: Springer Verlag.

Fialho, A., Da Costa, L., Schoenauer, M., and Sebag, M. (2008). Extreme value based adaptive operator selection. In *Parallel Problem Solving from Nature (PPSN X). Lecture Notes in Computer Science,* Vol. 5199 (pp. 175–184). Berlin: Springer. Available at http://hal.inria.fr/inria-00287355/en/

Fialho, A., Da Costa, L., Schoenauer, M., and Sebag, M. (2010). Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence,* Special issue on learning and intelligent optimization. DOI 10.1007/s10472-010-9213-y. Available at http://dx.doi.org/10.1007/s10472-010-9213-y

Fogel, D. (1992). Evolving artificial intelligence. PhD thesis, University of California at San Diego.

Fukunaga, A. (2008). Automated discovery of local search heuristics for satisfiability testing. *Evolutionary Computation,* 16(1):31–61.

Kendall, G., Cowling, P., and Soubeiga, E. (2002). Choice function and random hyperheuristics. In *Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL),* pp. 667–671.

Krasnogor, N. (1999). Coevolution of genes and memes in memetic algorithms. In A. Wu (Ed.), *Proceedings of the 1999 Genetic and Evolutionary Computation Conference Workshop Program*.

Krasnogor, N. (2002). Studies in the theory and design space of memetic algorithms. PhD thesis, University of the West of England.

Krasnogor, N. (2004). Self-generating metaheuristics in bioinformatics: The protein structure comparison case. *Genetic Programming and Evolvable Machines,* 5(2):181–201.

Krasnogor, N., and Gustafson, S. (2004). A study on the use of "self-generation" in memetic algorithms. *Natural Computing,* 3(1):53–76.

Krasnogor, N., and Smith, J. (2000). A memetic algorithm with self-adaptive local search: TSP as a case study. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, and H. G. Beyer (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000),* pp. 987–994.

Krasnogor, N., and Smith, J. (2001). Emergence of profitable search strategies based on a simple inheritance mechanism. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. M. Voight, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001),* pp. 432–439.

Krasnogor, N., Blackburne, B., Burke, E., and Hirst, J. (2002). Multimeme algorithms for protein structure prediction. In J. M. Guervos, P. Adamidis, J. L. Fernandez-Villacanas, H. P. Schwefel

(Eds.), *Proceedings of the 7th Conference on Parallel Problem Solving from Nature. Lecture Notes in Computer Science,* Vol. 2439, pp. 769–778.

Maturana, J., Fialho, Á., Saubion, F., Schoenauer, M., and Sebag, M. (2009). Extreme compass and dynamic multi-armed bandits for adaptive operator selection. In *IEEE Congress on Evolutionary Computation,* Trondheim, Norway.

Meuth, R., Lim, M., Ong, Y., and Wunsch, D. (2009). A proposition on memes and meta-memes in computing for higher-order learning. *Memetic Computing,* 1(2):85–100.

Moscato, P. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech Concurrent Computation Program Report 826. Pasadena, CA: Caltech.

Neri, F. (2007a). An adaptive multimeme algorithm for designing HIV multidrug therapies. *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* 4(2):264–278.

Neri, F. (2007b). Fitness diversity based adaptation in multimeme algorithms: A comparative study. In *IEEE Congress on Evolutionary Computation, CEC 2007,* pp. 2374–2381.

Nguyen, H. Q., Ong, Y. S., Lim, M. H., and Krasnogor, N. (2009a). Adaptive cellular memetic algorithms. *Evolutionary Computation,* 17(2):231–256.

Nguyen, Q. H., Ong, Y. S., and Lim, M. H. (2009b). A probabilistic memetic framework. *IEEE Transactions on Evolutionary Computation,* 13(3):604–623.

Ong, Y., and Keane, A. (2004). Meta-Lamarckian learning in memetic algorithms. *IEEE Transactions on Evolutionary Computation,* 8(2):99–110.

Ong Y., Lim, M., Zhu, N., and Wong, K. (2006). Classification of adaptive memetic algorithms: A comparative study. *IEEE Transactions on Systems, Man and Cybernetics, Part B,* 36(1):141–152.

Ong, Y. S., Lim, M. H., and Chen, X. (2010). Memetic computation—Past, present & future [Research Frontier]. *Computational Intelligence Magazine,* 5(2):24–31.

Ostermeier, A., Gawelczyk, A., and Hansen, N. (1994). A derandomized approach to self-adaptation of evolution strategies. *Evolutionary Computation,* 2(4):369–380.

Parker, G., and Blumenthal, H. (2004). Varying sample sizes for the co-evolution of heterogeneous agents. In *Proceedings of the Congress on Evolutionary Computation (CEC 2004),* pp. 766–771.

SATLIB. (2010). Online repository of satisfiability problems. Available at http://www.satlib.org

Schaffer, J., and Morishima, A. (1987). An adaptive crossover distribution mechanism for genetic algorithms. In J. Grefenstette (Ed.), *Proceedings of the 2nd International Conference on Genetic Algorithms and Their Applications,* pp. 36–40.

Schwefel, H. P. (1981). *Numerical optimisation of computer models*. New York: Wiley.

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

Smith, J. (2001). Modelling GAs with self-adaptive mutation rates. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. M. Voight, M. Gen, S. Sen, M. Dorigon, S. Pezeshk, M. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2011),* pp. 599–606.

Smith, J. (2002a). Co-evolution of memetic algorithms: Initial investigations. In J. M. Guervos, P. Adamis, J. L. Fernandez-Villacanas, and H. P. Schwefel (Eds.), *Proceedings of the 7th Conference on Parallel Problem Solving from Nature. Lecture Notes in Computer Science,* Vol. 2439, pp. 537–548.

Smith, J. (2002b). On appropriate adaptation levels for the learning of gene linkage. *Journal of Genetic Programming and Evolvable Machines,* 3(2):129–155.

Smith, J. (2003a). Co-evolving memetic algorithms: A learning approach to robust scalable optimisation. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, pp. 498–505.

Smith, J. (2003b). Parameter perturbation mechanisms in binary coded gas with self-adaptive mutation. In *Proceedings of Foundations of Genetic Algorithms, 7*, pp. 329–346.

Smith, J. (2003c). Protein structure prediction with co-evolving memetic algorithms. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, pp. 2346–2353.

Smith, J. (2004). The co-evolution of memetic algorithms for protein structure prediction. In W. Hart, N. Krasnogor, and J. Smith (Eds.), *Recent advances in memetic algorithms* (pp. 105–128). Berlin: Springer .

Smith, J. (2007a). Co-evolving memetic algorithms: A review and progress report. *IEEE Transactions in Systems, Man and Cybernetics, Part B*, 37(1):6–17.

Smith, J. (2007b). Credit assignment in adaptive memetic algorithms. In *Proceedings of GECCO, the ACM-SIGEVO Conference on Evolutionary Computation*, pp. 1412–1419.

Smith, J. (2010). Meme fitness and memepool sizes in coevolutionary memetic algorithms. *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8.

Smith, J., and Fogarty, T. (1996a). Adaptively parameterised evolutionary systems: Self adaptive recombination and mutation in a genetic algorithm. In H. M. Voigt, W. Ebeling, I. Rechenberg, and H. P. Schwefel (Eds.), *Proceedings of the 4th Conference on Parallel Problem Solving from Nature. Lecture Notes in Computer Science*, Vol. 1141. (pp. 441–450). Berlin: Springer.

Smith, J., and Fogarty, T. (1996b). Recombination strategy adaptation via evolution of gene linkage. In *Proceedings of the 1998 IEEE Conference on Evolutionary Computation*, pp. 826–831.

Smith, J., and Fogarty, T. (1996c). Self adaptation of mutation rates in a steady state genetic algorithm. In *Proceedings of the 1998 IEEE Conference on Evolutionary Computation*, pp. 318–323.

Smith, J., and Fogarty, T. (1997). Operator and parameter adaptation in genetic algorithms. *Soft Computing*, 1(2):81–87.

Stone, C., and Smith, J. (2002). Strategy parameter variety in self-adaption. In W. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. Potter, A. Schultz, J. Miller, E. Burke, and N. Jonoska (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, pp. 586–593.

Thierens, D. (2007). Adaptive strategies for operator allocation. In F. Lobo, C. Lima, and Z. Michalewicz (Eds.), *Parameter setting in evolutionary algorithms*, pp. 77–90.

Ting, C., Zeng, W., and Lin, T. (2010). Linkage discovery through data mining. *Computational Intelligence Magazine*, 5:10–13.

Whitacre, J. M., Pham, T. Q., and Sarker, R. A. (2006). Credit assignment in adaptive evolutionary algorithms. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 1353–1360.

Whitacre, J. M., Sarker, R. A., and Pham, Q. T. (2008). The self-organization of interaction networks for nature-inspired optimization. *IEEE Transactions on Evolutionary Computation*, 12(2):220–230.

Wiegand, R., Liles, W., and Jong, K. D. (2001). An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. M. Voight, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2011)*, pp. 1235–1245.