Operator and Parameter Adaptation in Genetic Algorithms

J.E. Smith
Intelligent Computer Systems Centre
Faculty of Computer Studies & Mathematics
Frenchay Campus
University of the West of England
Bristol, BS16 1QY
jim@btc.uwe.ac.uk

T.C.Fogarty
Department of Computer Studies
Napier University
Craiglockhart Campus
219 Colinton Road
Edinburgh EH14 1DJ
T.Fogarty@dcs.napier.ac.uk

Abstract

Genetic Algorithms are a class of powerful, robust search techniques based on genetic inheritance and the Darwinian metaphor of "Natural Selection". These algorithms maintain a finite memory of individual points on the search landscape known as the "population". Members of the population are usually represented as strings written over some fixed alphabet, each of which has a scalar value attached to it reflecting its quality or "fitness". The search may be seen as the iterative application of a number of operators, such as selection, recombination and mutation, to the population with the aim of producing progressively fitter individuals.

These operators are usually static, that is to say that their mechanisms, parameters, and probability of application are fixed at the beginning and constant throughout the run of the algorithm. However there is an increasing body of evidence that not only is there no single choice of operators which is optimal for all problems, but that in fact the optimal choice of operators for a given problem will be time-variant i.e. it will depend on such factors as the degree of convergence of the population. Based on theoretical and practical approaches, a number of authors have proposed methods of adaptively controlling one or more of the operators, usually invoking some kind of "meta-learning" algorithm, in order to try and improve the performance of the Genetic Algorithm as a function optimiser.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

In this paper we describe the background to these approaches, and suggest a framework for their classification based on the learning strategy used to control them, and what facets of the algorithm are susceptible to adaptation. We then review a number of significant pieces of work within this context, and draw some conclusions about the relative merits of various approaches and promising directions for future work.

1: Introduction

1.1: A Background to Adaptation in Genetic Algorithms

Genetic Algorithms [Holland (1975)] are a class of population based randomised search techniques which are increasingly widely used in a number of practical applications. Typically these algorithms maintain a number of potential solutions to the problem being tackled, which can be seen as a form of working memory - this is known as the population. Iteratively new points in the search space are generated for evaluation and are optionally incorporated into the population. Attached to each point in the search space will be a unique fitness value, and so we can usefully envisage the space as a "fitness landscape". It is the population which provides the algorithm with its power by providing a means of defining a non-uniform probability distribution function (p.d.f.) governing the generation of new points on the landscape. This p.d.f. reflects possible interactions between points in the population, arising from the "recombination" of partial solutions from two (or more) members of the population (parents). This contrasts with the globally uniform distribution of blind random search, or the locally uniform distribution used by many other stochastic algorithms such as simulated annealing and various hill-climbing algorithms.

The genetic search may be viewed as the iterated application of two processes. Firstly *Generating* a new set of candidate points. This is done probabalistically according to the p.d.f. defined by the action of the chosen reproductive operators (recombination and muta-

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

tion) on the original population. Secondly *Updating* the algorithms working memory (population). This usually done by evaluating each new point, then applying some kind of selection algorithm to the union of these and the parent population in order to produce a new one.

The efficiency of the algorithm can thus be seen to depend on two factors, namely the maintenance of a suitable working memory, and quality of the match between the p.d.f. generated and the landscape being searched. The first of these factors will depend on the choices of population size and selection algorithm. The second will depend on the action of the reproductive operators and their associated parameters on the current population.

Naturally, a lot of work has been done on trying to find suitable choices of operators and their parameters which will work over a wide range of problem types. The first major study [DeJong (1975)] identified a suite of test functions and proposed a set of parameters which it was hoped would work well across a variety of problem types. However later studies using a "meta-ga" to learn suitable values [Grefenstette (1986)] or using exhaustive testing [Schaffer et al. (1989)] arrived at different conclusions. Meanwhile theoretical analysis on optimal population sizes [Goldberg (1985)] started to formalise the (obvious?) point that the size of the population on the basis of which decisions could be reliably made depends on the size of the search space.

The next few years of research saw a variety of new operators proposed, some of which (e.g. Uniform Crossover [Syswerda (1989)]) forced a reappraisal of the Schema Theorem (the decision theoretic background formulated in Holland's early work). This led to the focusing on two important concepts.

The first of these, "Crossover Bias" [Eshelman et al. (1989)], refers to the differing ways

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

in which the p.d.f.'s arising from various crossover operators maintain information about the fitness of hyperplanes of high estimated fitness. This is a function of the suitability of the p.d.f. induced by the recombination operator to the landscape induced by the problem encoding. The empirical findings have been confirmed by more formal analysis on the relative merits of various recombination mechanisms [DeJong and Spears (1990,1992), Spears and DeJong (1991)].

The second concept was that of "Safety Ratios" [Schaffer and Eshelman (1991)] This is the probability that a new point generated by the application of reproductive operators would be fitter than its parent(s). These ratios were shown empirically to be different for the various reproductive operators, and also to change over time. Again these reflect the match between the p.d.f.s induced by given operators on the current population to the fitness contours of the landscape.

These considerations, when coupled with interactions with other Evolutionary Algorithm communities who were already using adaptive operators (e.g. the (1+1) [Rechenberg (1973)] and (m l) [Schwefel (1977), Schwefel (1981)] Evolutionary Strategies) has led to an ever increasing interest in the possibilities of developing algorithms which are able to adapt one or more of their operators or parameters over time. The aim is to match the p.d.f. induced by the algorithm to the fitness landscape.

As soon as we allow for variation in the processes of *Generating* or *Updating* we effectively increase the size of the problem since we are now not only traversing the problem space but also the space of all variants of the basic algorithm. This traversal may take several forms, depending on the scope of change allowed and the nature of the learning algorithm. It may vary from the simple time-dependant decrease in the value of a single

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

parameter according to some fixed rule, e.g [Fogarty (1989)], to a complex path which potentially occupies any position in the latter space and which is wholly governed by the *Updating* process, e.g. [Smith and Fogarty (1996a)].

1.2: A Framework for Classifying Adaptation in Genetic Algorithms

In this paper we shall construct a framework for categorising various proposed methods of incorporating adaptation into Genetic Algorithms. This is based on three principles, namely *What* is being adapted (operators, parameters etc.), the *Scope* of the adaption (i.e. does it apply to all the population, just individual members or just sub-components) and the *Basis* for change (e.g. externally imposed schedule, fuzzy logic etc.). In the following sections we will describe these principles in more depth, and within the various categories we shall describe some examples of each types of adaptation.

2: *What* is being Adapted?

As has been described above, the genetic algorithm may be viewed as the iterated application of two processes; *Generating* new points in the landscape (via probabalistic application of recombination and/or mutation operators to the previous population), and *Updating* (via selection and resizing) to produce a new population based on the new set of points created and (possibly) the previous population.

In general most of the proposed variants of the simple genetic algorithm only act on a single operator, and furthermore it is true to say that most work has concentrated on the reproductive operators i.e. recombination and mutation. Whilst considerable effort has been expended on the question of what proportion of the population should be replaced at any given iteration, the mechanisms for updating the population, once chosen, tend to be static.

Broadly speaking most adaptive algorithms work with the settings of either a Generational GA (GGA) or a “Steady State” GA (SSGA) [Whitley and Kauth (1988)] which rep-

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

resent the two extremes of i) generating an entirely new population and discarding the previous population in the selection process or ii) only generating and replacing a single individual at each iteration. Much theoretical analysis of GA's has focused on the twin goals of *exploration* of new regions of the search space and *exploitation* of previously learned good regions (or hyperplanes) of the search space. The two terms may be explained by considering how the p.d.f.s governing generation of new points change over algorithmic time, remembering that the p.d.f.s are governed jointly by the actions of the reproduction and selection operators. An explorative algorithm is one in which a relatively high probability is assigned to regions as yet unvisited by the algorithm, whereas an exploitative algorithm is one in which the p.d.f. represents rather accumulated information about relatively fitter regions and hyperplanes of the search space. Thus the p.d.f. of an explorative algorithm will change more rapidly than that of an exploitative one.

If reproductive operators are applied which produce offspring that are unlike their parents, then the updating mechanisms of GGA's and SSGA's may be seen as favouring exploration and exploitation respectively. However the extent to which offspring differ from their parents is governed not simply by the type of operator applied, but also by the probability of its application as opposed to simply reproducing the parents. Thus for a given population and reproductive operator, we can tune the shape of the induced p.d.f. between the extremes of exploration and exploitation by altering the probability of application regardless of the updating mechanism.

It is this last point which has led to a focusing on the adaptation of the reproductive operators, since by changing the amount of disruption induced by the operators it is possible to indirectly adapt the proportion of the population which is replaced at each iteration

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

whilst using simple and efficient selection mechanisms. This allows the tuning of the updating mechanism to particular application characteristics e.g. the possibility of parallel evaluations etc.

We can distinguish a number of different strands of research by considering what it is that the algorithms adapt.

In some ways the simplest class are those algorithms which use a fixed set of operators and adapt the probability of application of those operators. Perhaps the two best known early examples of work of this type are Fogarty's use of a time-dependant probability of mutation [Fogarty (1989)] and Davis' use of varying probabilities of application of a few simple operators (uniform crossover, averaging crossover, mutation, "big creep" and little creep") depending on their performance over the last few generations [Davis (1989)]. Many later authors have proposed variants on this approach, using a number of different learning methods as will be seem later. A similar approach which changes the population updating mechanism by changing the selection pressure over time can be seen in the Grand Deluge Evolutionary Algorithm [Rudolph and Sprave (1995)].

An extension of this approach can be seen in algorithms which maintain distinct sub-populations, each using different sets of operators and parameters. A common approach is to use this approach with a "meta-ga" specifying the parameters for each sub-population- e.g. [Grefenstette (1986), Kakuza et al. (1992), Friesleben and Hartfelder (1993)] to name but three. Although the searches in different (sub)populations may utilise different operators, we can view each as having the full set available, but with zero probability of applying most, hence we group these in the first category

A second class of adaptive GA's can be distinguished as changing the actual action of

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

the operator(s) over time. An early example of this was the "Punctuated Crossover" mechanism [Schaffer and Morishima (1987)] which added extra bits to the representation to encode for crossover points. These were allowed to evolve over time to provide a 2 parent N-point recombination mechanism where N was allowed to vary between zero and the length of the string. The LEGO and related APES mechanisms [Smith and Fogarty (1995, 96a, 96b)] evolve the "units of heredity" which determine recombination and mutation mechanisms, through the use of genetic encoded "links" between loci on the representation. Within this category we can also place algorithms which alter the mechanisms governing the updating of the working memory by changing its size (e.g.[Smith (1993), Smith and Smuda (1995), Arabas et al. (1994), Hinterding et al. (1996)]).

Finally we should mention an alternative approach, which is to alter the representation of the problem itself: this can be viewed as an attempt to make the landscape suit the p.d.f.s induced by the operators rather than vice-versa. Goldberg's work on "Messy GA's" ([Goldberg et al. (1989)] and many subsequent publications) is based on finding appropriate linkages between genes, using a "floating" representation where the order of the variables is not fixed. Since the "cut and splice" recombination operator tends to keep together adjacent genes, this can be seen as moulding the landscape to suit the operators. Similarly Schaefer's ARGOT [Schaefer (1987)] strategy adaptively resizes the representation according to global measures of the algorithm's success. More recently work on co-evolving representations [Paredis (1995)] can be seen in this restructuring light, as can the Adaptive Penalty Functions of [Eiben and van der Hauw (1996)].

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

3: What is the *Scope* of the Adaptation?

We shall adopt the terminology of [Angeline (1995)] to define three distinct levels at which adaptation can occur in evolutionary algorithms. Population-level adaptations make changes which affect the p.d.f. contribution from each member of the current population in the same way. Individual-level adaptations make changes which affect the p.d.f. contribution from each member of the population separately, but apply uniformly to each of the components of the individuals' representation. At the finest level of granularity are Component-level adaptations, where the p.d.f. contributions from each component of each member may be changed individually.

3.1: Population-Level Adaptation.

Recall from above that in most cases these operators in a GA are static (that is to say that their forms and parameters are fixed) and apply uniformly to the whole population i.e. they are global. Population-level adaptation algorithms can be typified as using a fixed set of global operators, but allowing their parameters to vary over time. The most important parameter is of course the probability of application. The various "meta-ga" algorithms mentioned above, and the competing sub-populations of the breeder GA e.g. [Schlierkamp-Voosen and Muhlenbein (1994)] belong firmly to this level.

There are a number of theoretical results which provide support for this kind of approach, e.g. regarding the time-variance of the optimal mutation rate [Muhlenbein (1992), Hesser and Manner (1991)], as well as Schaffer & Eshelman's experimental results regarding the time dependencies of "safety ratios" for mutation and crossover.

In [Fogarty (1989)] an externally defined form is used to reduce the mutation rate over time, in a similar fashion to the "cooling schedules" used in Simulated Annealing. However a more common approach is to adjust one or more parameters dynamically in accordance

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

with the performance of the algorithm or some measured quantity of the population.

A well known, and popular approach (e.g. [Davis (1989), Corne et al. (1994), Julstrom (1995)]) is to keep statistics on the performance of offspring generated by various reproductive operators relative to their parents. Periodically "successful" operators are rewarded by increasing their probability of application relative to less successful operators. This approach requires extra memory, since it is usually found necessary to maintain family trees of the operators which led to a given individual. This is done in order to escape from credit allocation problems.

Other authors have proposed control strategies based on simpler measures. Eshelman and Schaffer use the convergence of the population to alter the thresholds governing incest prevention, and the time spent without improvement to govern the probability of restarting the GA using vigorous mutation [Eshelman and Schaffer (1991)]. This latter is similar to many strategies for tracking changing environments where a drop in the performance of the best member of the current generation is used to trigger a higher rate of mutation e.g. [Cobb and Grefenstette (1993)].

In [Lee and Takagi (1993)] fuzzy rules are learned and used to control various parameters based on the relative performance of the best, worst and mean of the current population. This concept of observing the fitness distribution of the population and then altering parameters according to a set of rules is also used in [Lis (1996)] where the mutation rate is altered continuously in order to keep a fitness distribution metric - the "population dispersion rate" within a desired range.

A more complex approach, and one which at first appears to belong to the component level is that of [Sebag and Schoenauer (1994)] who maintain a library of "crossover masks"

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

which they label as good or bad. Inductive learning is used to control the application of crossover and the periodic updating of mask strengths. However the rules learnt are applied uniformly to the whole population and so this belongs firmly in the category of population-level adaptation.

Perhaps the most obvious method of achieving population-level adaptation is to dynamically adjust the size of the population itself by creating or removing members according to some global measurement. Two approaches have been recently reported. In [Smith (1993), Smith and Smuda (1995)] the population is adjusted according to estimated schema fitness variance, whereas in [Hinterding et al. (1996)] three populations of different sizes are maintained, with periodic resizing according to their relative successes.

Finally we should point out that many of the proposed methods of adapting representations mentioned above plainly fall into this category.

3.2: Individual Level Adaptation

An alternative approach to adaptation is centred on consideration of the individual members of the population rather than the ensemble as a whole. Thus as a simple example, a global level adaptation may vary the probability of mutation for the whole population, whereas an individual level algorithm might hold a separate mutation probability for each member of the population. If we consider the pd.f. governing generation as the sum of the contributions from each member, then population level changes affect the way in which the contributions are determined uniformly, whereas individual level adaptations affect the p.d.f. contributions for each member separately.

A frequently claimed justification for this approach is that it allows for the learning of different search strategies in different parts of the search space. This is based on the not unreasonable assumption that in general search space will not be homogeneous, and that dif-

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

ferent strategies will be better suited to different kinds of sub-landscape.

As a crude metaphor, imagine a blind robot, equipped only with an altimeter, dropped at random on the earth and trying to reach the highest point it can. If on a large flat region, it would be better employed using a wider search (more uniform p.d.f. i.e. exploration), whereas if dropped by chance in the Himalayas such as search might rapidly lead it out of the mountain range. In the latter case a much more local search (i.e. exploitation) would be far preferable.

Algorithms at this level have been proposed which fall into a number of different categories in terms of the basis and scope of adaptation.

Perhaps the most popular class are algorithms which encode some parameters for an operator into the individual and allow these to evolve, using the updating process itself as the basis for learning. An early example of this was the "Punctuated Crossover" mechanism of [Schaffer and Morishima (1987)]. This added a mask to the representation which was used to determine crossover points between two parents during recombination, and which was evolved along with the solutions. The results reported were encouraging, but this may have been due to the high number of crossover points evolved (compared to the algorithm used for comparison). In [Levenick (1995)] a similar mechanism is investigated, but with the added bits coding for changes in crossover probability at those points rather than deterministic crossing.

An alternative method of controlling recombination strategies was used in [Spears (1995)] where a single bit was used to decide whether two-point or uniform crossover would be used when an individual reproduced. This differs from the above mechanisms in that they allow the form of the operator itself to change, whereas this effectively associates

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

two operators with each individual and associates probabilities of 0 or 100% of application with each. Interestingly, this was compared with a population level mechanism where the relative proportion of bits encoding for the two operators was used to make a global decision about operator probabilities. Although these two alternatives should provide the same ratio of usage of the two operators, the latter mechanism was found to work far less well, suggesting that there is indeed a merit in attaching reproductive strategies to particular individuals. Certainly the two methods will yield different p.d.f.s for the next generation of points.

Perhaps more widely used has been the "borrowing" of the idea of self adaptive mutation rates from Evolutionary Strategies. This addition of extra bits to the genome to code for the mutation rate for that individual was first investigated for GAs in [Back (1992)] when it was found that the rates evolved were close to the theoretical optimum for a simple problem (providing that a sufficiently rigorous selection mechanism was used). The second level of adaptation in [Hinterding et al. (1996)] uses a similar encoding to control the standard deviation of the Gaussian used to mutate the genes in a real-coded G.A. In [Smith and Fogarty (1996c)] self adaptive mutation was translated into a Steady State algorithm, with the addition of a (l,1) "inner-ga". This was necessary to provide the necessary selection pressure for self adaptation, but also provides a kind of local search as well.

However not all algorithms using adaptation at the individual level rely on endogenous control and self-adaptation. As with population level adaptations, a number of other mechanisms can be used to control the behaviour of individuals.

A typical example is found in [Srinivas and Patnaik (1994)] where the probabilities of applying mutation or crossover to an individual depend on its relative fitness, and the de-

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

gree of convergence (of fitnesses) of the population. This fitness-dependant control of the p.d.f. contribution from each individual is used to control the global balance of exploitation and exploration, so that the information in relatively fitter individuals is exploited whereas exploration is achieved by associating more uniform p.d.f.s with the less fit members. Another approach based on relative local fitness in a structured population uses a pattern of rules to control the reproduction strategy (replication, crossover or mutation) based on metaphors of social behaviour [Mattfeld et al. (1994)]. Similarly the Grand Deluge Evolutionary Algorithm of [Rudolph and Sprave (1995)] adapts the "acceptance" threshold governing the updating process separately for each point based on relative local fitness over a period of time.

Similarly measures of relative fitness and convergence are used to determine the "lifespan" given to a newly created individual in [Arabas et al. (1994)]. This novel approach controls the contents and size of the working memory by assigning a fixed lifetime to each individual after which it is removed from the memory. This is one of the very few algorithms reported providing dynamic population sizing, and the results reported suggest that this is a promising line of research.

3.3: Component-Level Adaptation

This is the finest level of granularity for adaptation: here algorithms allow different reproduction strategies for different parts of the problem representation. Again we could consider these ideas to have be based in the Evolutionary Strategies community where techniques have progressed over the years from a single parameter controlling the mutation step size for the population, through individual step-sizes for each member, to having a separate mutation step size encoded for each component being optimised (remember Evolu-

tionary Strategies work with a vector of real numbers).

We can consider this as a means of allowing the focus of search to be directed, and the principal advantage is that it allows for a much finer degree of "tuning" of the p.d.f. contribution associated with each individual.

This approach was tried in the Self-Adaptive mutation mechanism of [Back (1992)] and compared with individual level adaptation. The results suggested that in certain circumstances the results were advantageous, but that on other landscapes the added learning overheads associated with all the extra mutation parameters slowed the search down. The conclusions seemed to be that such a mechanism could be very effective if the components were properly defined i.e. the level of granularity was chosen suitably.

An attempt to solve some of these problems can be seen in the LEGO mechanisms [Smith and Fogarty (1995, 96b)]. These are somewhat similar to the "Punctuated Crossover" algorithms of Schaffer and Morishima in adding extra bits to the representation to determine whether two adjacent genes may be broken by crossover. However the emphasis is different, in concentrating on finding blocks of co-evolved linked genes. The recombination mechanism differs in that blocks of genes may be chosen from the whole population (rather than just two parents) when a new individual is formed.This evolution of linkage has to be seen therefore as an adaptation of recombination strategy at the component level, since individuals have no meaning in the context of parents other than as contributing to a "genepool" from which new individuals are assembled. This emphasis on evolving successful components rather than individuals is taken further in the APES algorithm [Smith and Fogarty (1996a)] which includes component level adaptation of mutation rates by attaching a mutation rate for each block, which is also self-adapted.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

4: **What is the Basis** for Adaptation?

We now turn to perhaps the most important distinction, namely the basis on which adaptation is carried out. This in turn hinges on two factors, firstly the evidence upon which adaptation is carried out, and secondly the rules or algorithm which define how changes are effected.

More formally, what we are concerned with at this point is the way in which features of the Generating and/or Updating processes are changed in an attempt to match the p.d.f.s governing the transition between populations to the topography of the search landscape.

If we consider an "algorithmic space" within which we can locate any given variant of the GA we can thus define adaptation as describing a traversal of this algorithmic space, in much the same way as the GA's population traverses the search space induced by the problem representation. For population level changes this trajectory will be a single path. For individual level adaptations, we must consider the path of a cloud, as each member follows a slightly different course. This cloud is composed of a number of traces which may appear and disappear under the influence of selection. In either case, what we arte concerned with is the mechanism that defines the trajectory.

Immediately, we can draw an important distinction between two types of algorithm. These have been labelled as "Tightly Coupled vs. Uncoupled" by Spears, or "Empirical vs. Absolute" update rules by Angeline. Essentially in the first type of algorithm, an external mechanism is used to determine the trajectory, whereas in the second, the selection mechanism of the GA itself is used to determine the trajectory. This latter approach is more commonly known as Self-Adaptation.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

To clarify this point, in all algorithms a set of evidence is considered, on the basis of which the trajectory of the algorithm is decided. In Self-Adaptive algorithms the evidence is simply the relative fitness of the individual(s) with which that strategy is associated, and the p.d.f. of the trajectory is defined by the selection mechanism of the algorithm itself. In other adaptive algorithms, the evidence generally takes the form of statistics about the performance of the algorithm, such as the fitness distribution of the population, "family trees", or simply the amount of evolutionary time elapsed. The important factor is that the mechanism used to generate the new strategies based on this evidence is externally provided in the form of a learning algorithm or a set of fixed rules.

We have already noted many Self-Adaptive algorithms, and their reported success is perhaps not surprising in the light of the considerable research into these topics in the fields of Evolution Strategies and Evolutionary Programming. The rationale is essentially twofold, firstly that if strategies are forced to compete through selection, then what better proof of the value of a strategy than its continued existence in the population? Secondly, and perhaps less tritely, we can observe that the algorithmic space being searched has an unknown topography, but is certainly extremely large, very complex and highly epistatic, since there is a high degree of dependency between operators. Evolutionary Algorithms have repeatedly been shown to be good at searching such spaces.

This same rationale is used for many of the "meta-ga" strategies, the "competing subpopulations" of the Breeder GA [Schlierkamp-Voosen and Muhlenbein (1994)], and the population sizing adaptation described in [Hinterding et al. (1996)]. However it is more correct to describe these as Uncoupled Adaptations since the strategies are not coded for directly in the populations, and the evidence for change is the relative performance of (sub)

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

populations of individuals in a given time period.

We can in fact extend this idea of relative performance to draw a second distinction between all kinds of adaptive algorithms on the basis of the evidence that they consider.

In one camp are all those algorithms which take as their evidence differences in the relative performance of strategies. These include Self-Adaptive and meta-GAs as just described. Those algorithms which adjust operator probabilities based on records of performance, such as those of Whitley, Julstrom and Corne et al. described above fall into this category as their probabalistic nature allows the concurrent testing of multiple strategies (the difference being that they use predefined rules to adjust the strategies). Also into this category fall those algorithms which draw conclusion based on observations of strategy performance such as Sebag and Schoenauer's inductively learned Crossover Masks and White and Oppacher's Automata Controlled Crossover.

Into the other camp fall those algorithms which adapt strategies based on empirical evidence which is not directly related to the strategy followed. This might be convergence statistics (in terms of the population fitness, or the allele distribution), observed relative schema fitnesses etc. All of these algorithms feature Uncoupled Adaptation, and although some use simple rules e.g. previously learned Fuzzy Rule-Sets [Lee and Takagi (1993)] or predefined schedules [Fogarty (1989)], most base the derived trajectory on an attempt to fulfil some criterion or maintain a population statistic. Examples of this latter are the "Population Dispersal" metric of [Lis (1996)], and "Schema Fitness Variance" [Smith and Smuda (1995)].

5: Discussion

In the discussion above we categorised the algorithms described according to what

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

features of the algorithm are susceptible to adaptation, and the granularity at which this is done. We have drawn a further important distinction according to the basis for change. This took two forms, namely the type of evidence used as input to the strategy-deciding algorithm, and that algorithm itself. There are a number of possible ways of representing this categorisation. In Figure 1, a simplified taxonomic tree is shown to illustrate the main branches of Adaptive Genetic Algorithms (algorithms are referred to by their first authors). Algorithms involving a restructuring of the search space via representation changes are not shown, but these are relatively uncommon. However this diagram illustrates the point that the majority of the work published in this field can be distinguished by common strands of thought.

The nature of the field, and the absence of a standard test suite of problems make it impossible to compare algorithms on the basis of reported results and declare a universal "winner". Indeed the very considerations that drive research into adaptive algorithms make such statements meaningless. It is however possible to draw a few conclusions.

Firstly, it seems that in general those algorithms based on observing the relative performance of different strategies appear to be most effective. This seems to follow naturally from the fact that GA theory is currently not sufficiently advanced either to permit the specification of suitable goals in terms of other metrics, or (more importantly) how to achieve them. It is ironic that perhaps the most widely quoted paper on adaptive strategies, the externally defined time-decreasing mutation rate of [Fogarty (1989)] is also one of the most wildly misquoted works in the field, since this was concerned specifically with initially converged populations.

Secondly, there appears to be a distinct need for the maintenance of sufficient diversity

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

with the population(s). It is the experience of several authors working with adaptive recombination mechanisms that convergence makes the relative assessment of different strategies impossible. Variety within the population is vital as the driving force of selective pressure in all Evolutionary Algorithms, and will be doubly so in Self-Adaptive algorithms. This is less of a problem for algorithms manipulating mutation rates as mutation is generally a force for increased diversity.

Thirdly, there are powerful arguments and empirical results for pitching the adaptation at an appropriate level. The success of individual level adaptive reproduction schemes appears convincing, and there is promise in the various methods proposed for identifying suitable "components" via linkage analysis which would allow adaptation at an appropriate level. However as Angeline points out, and Hinterding demonstrates, there is scope for adaptation to occur at a variety of levels with the GA

Finally we must consider the nature of adaptation in terms of searching the space of possible configurations for the GA This space is one about which little is known, and needless to say the landscape will depend on what is being optimised (e.g. best solution in a given time, mean performance etc.). However it is the fact that little is known about it which makes it so similar to those problem landscapes which we are interested in exploring. All the arguments used in favour of adaptive GA's in the first section of this paper would appear to apply equally to the search of this space.

It is for this reason that the author's personal experience leads them to believe that Self-Adaptation at a variety of levels represents the most promising area for future research.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

6: References

Angeline, P. J. (1995). Adaptive and self-adaptive evolutionary computations. Computational Intelligence, pp. 152–161. IEEE Press.

Arabas, J., Michalewicz, Z., and Mulawka, J. (1994). Gavaps - a genetic algorithm with varying population size. Proceedings of the First IEEE International Conference on Evolutionary Computing, pp. 73–78. IEEE Press.

Back, T. (1992). Self adaptation in genetic algorithms. Varela, F. and Bourgine, P. (eds), Proceedings of the First European Conference on Artificial Life, pp. 263–271. MIT Press.

Cobb, H. G. and Grefenstette, J. J. (1993). Genetic algorithms for tracking changing environments. Forrest, S. (ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 523–530. Morgan Kaufmann.

Corne, D., Ross, P., and H.-L.Fang (1994). Fast practical evolutionary timetabling. Technical Report GA-research Note 7, University of Edinburgh Dept. of Artificial Intelligence.

Davis, L. (1989). Adapting operator probabilities in genetic algorithms. J.J.Grefenstette (ed.), Proceedings of the Third International Conference on Genetic Algorithms, pp. 61–69. Morgan Kaufmann.

DeJong, K. (1975). An Analysis of the Behaviour of a Class of Genetic Adaptive Systems. PhD thesis, University of Michigan.

DeJong, K. A. and Spears, W. M. (1990). An analysis of the interacting roles of population size and crossover in genetic algorithms. Schwefel, H.-P. and Manner, R. (eds), Proceedings of the First Conference on Parallel Problem Solving from Nature, pp. 38–47. Springer Verlag.

DeJong, K. A. and Spears, W. M. (1992). A formal analysis of the role of multi-point crossover in genetic algorithms. Annals of Mathematics and Artificial Intelligence, 5(1):1–26.

Eiben, A. E. and van der Hauw, J. (1996). Graph colouring with adaptive genetic algorithms. Technical Report 96-11, Department of Computer Science, Leiden University.

Eshelman, L. J., Caruana, R. A., and Schaffer, J. D. (1989). Biases in the crossover landscape. Schaffer, J. D. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, pp. 10–19. Morgan Kaufmann.

Eshelman, L. J. and Schaffer, J. (1991). Preventing premature convergence in genetic algorithms by preventing incest. Booker, L. B. and Belew, R. (eds), Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 115–122. Morgan Kaufmann.

Fogarty, T. C. (1989). Varying the probablity of mutation in the genetic algorithm. Schaf-

fer, J. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, pp. 104–109. Morgan Kaufmann.

Friesleben, B. and Hartfelder, M. (1993). Optimisation of genetic algorithms by genetic algorithms. Albrecht, R., Reeves, C., and Steele, N. (eds), Artifical Neural Networks and Genetic Algorithms, pp. 392–399. Springer Verlag.

Goldberg, D. E. (1985). Optimal initial population size for binary-coded genetic algorithms. Tcga report no. 85001, University of Alabama, Tuscaloosa, US.

Goldberg, D. E., Korb, B., and Deb, K. (1989). Messy genetic algorithms: Motivation, analysis, and first results. Complex Systems, 3(5):493–530.

Grefenstette, J. J. (1986). Optimisation of control parameters for genetic algorithms. Transaction on Systems, Man and Cybernetics, 16(1):122–128.

Hesser, J. and Manner, R. (1991). Towards an optimal mutation probablity in genetic algorithms. Schwefel, H.-P. and Manner, R. (eds), Proceedings of the First Conference on Parallel Problem Solving from Nature, pp. 23–32. Springer Verlag.

Hinterding, R., Michalewicz, Z., and Peachey, T. (1996). Self adaptive genetic algorithm for numeric functions. Voigt, H., Ebeling, W., Rechenberg, I., and Schwefel, H.-P. (eds), Proceedings of the Fourth Conference on Parallel Problem Solving from Nature, pp. 420–429. Springer Verlag.

Holland, J. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Julstrom, B. A. (1995). What have you done for me lately?: Adapting operator probabilities in a steady-state genetic algorithm. Eshelman, L. J. (ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 81–87. Morgan Kaufmann.

Kakuza, Y., Sakanashi, H., and Suzuki, K. (1992). Adaptive search strategy for genetic algorithms with additional genetic algorithms. Männer, R. and Manderick, B. (eds), Proceedings of the Second Conference on Parallel Problem Solving from Nature, pp. 311–320. Elsevier Science.

Lee, M. and Takagi, H. (1993). Dynamic control of genetic algorithms using fuzzy logic techniques. Forrest, S. (ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 76–83. Morgan Kaufmann.

Levenick, J. R. (1995). Megabits: Generic endogenous crossover control. Eshelman, L. J. (ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 88–95. Morgan Kaufmann.

Lis, J. (1996). Parallel genetic algorithm with dynamic control parameter. Proceedings of the Third IEEE International Conference on Evolutionary Computing, pp. 324–329. IEEE Press.

Mattfeld, D., Kopfer, H., and Bierwirth, C. (1994). Control of parallel population dynamics by social-like behaviour of ga-individuals. Davidor, Y. (ed.), Proceedings of the Third Conference on Parallel Problem Solving from Nature, pp. 16–25. Springer Verlag.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"

Muhlenbein, H. (1992). How genetic algorithms really work : 1. mutation and hillclimbing. Manner, R. and Manderick, B. (eds), Proceedings of the Second Conference on Parallel Problem Solving from Nature, pp. 15–25. Elsevier Science.

Paredis, J. (1995). The symbiotic evolution of solutions and their representations. Eshelman, L. J. (ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 359–365. Morgan Kaufmann.

Rechenberg, I. (1973). Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Stuttgart, Frommann-Holzboog.

Rudolph, G. and Sprave, J. (1995). A cellular genetic algorithm with self adjusting acceptance threshold. Proceedings of the 1st IEE/IEEE International Conference on Genetic Algorithms in Engineering: Innovations and Applications, pp. 365–372. IEE.

Schaefer, C. (1987). The argot system: Adaptive representation genetic optimising technique. J.J.Grefenstette (ed.), Proceedings of the Second International Conference on Genetic Algorithms, pp. 50–58. Lawrence Erlbaum.

Schaffer, J. and Morishima, A. (1987). An adaptive crossover distribution mechanism for genetic algorithms. J.J.Grefenstette (ed.), Proceedings of the Second International Conference on Genetic Algorithms, pp. 36–40. Lawrence Erlbaum.

Schaffer, J. D., Caruana, R. A., Eshelman, L. J., and Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimisation. Schaffer, J. D. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, pp. 51–60. Morhan Kaufmann.

Schaffer, J. D. and Eshelman, L. J. (1991). On crossover as an evolutionarily viable strategy. Belew, R. and Booker, L. (eds), Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 61–68. Morgan Kaufmann.

Schlierkamp-Voosen, D. and Muhlenbein, H. (1994). Strategy adaptation by competing subpopulations. Davidor, Y. (ed.), Proceedings of the Third Conference on Parallel Problem Solving from Nature, pp. 199–209. Springer Verlag.

Schwefel, H. P. (1977). Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie, volume 26 of ISR. Basel/Stuttgart, Birkhaeuser.

Schwefel, H. P. (1981). Numerical Optimisation of Computer Models. New York, John Wiley and Sons.

Sebag, M. and Schoenauer, M. (1994). Controlling crossover through inductive learning. Davidor, Y. (ed.), Proceedings of the Third Conference on Parallel Problem Solving from Nature, pp. 209–218. Springer Verlag.

Smith, J. and Fogarty, T. (1996a). Adaptively parameterised evolutionary systems: Self adaptive recombination and mutation in a genetic algorithm. Voigt, Ebeling, Rechenberg, and Schwefel (eds), Proceedings of the Fourth Conference on Parallel Problem Solving from Nature, pp. 441–450. Springer Verlag.

Smith, J. and Fogarty, T. (1996b). Recombination strategy adaptation via evolution of gene linkage. Proceedings of the Third IEEE International Conference on Evolutionary

Computing, pp. 826–831. IEEE Press.

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

Smith, J. and Fogarty, T. C. (1995). An adaptive poly-parental recombination strategy. Fogarty, T. C. (ed.), Evolutionary Computing 2, pp. 48–61. Springer Verlag.

Smith, R. E. (1993). Adaptively resizing populations: An algorithm and analysis. Technical Report TCGA Report # 93001, University of Alabama, Box 870278, Tuscaloosa, Alabama 35487.

Smith, R. E. and Smuda, E. (1995). Adaptively resizing populations: Algorithm, analysis and first results. Complex Systems, 9(1):47–72.

Spears, W. M. (1995). Adapting crossover in evolutionary algorithms. McDonnel J.R., Reynolds, R.G., and Fogel, D.B.. (eds), Proceedings of the Fourth Annual Conference on Evolutionary Programming, pp. 367–384. MIT Press.

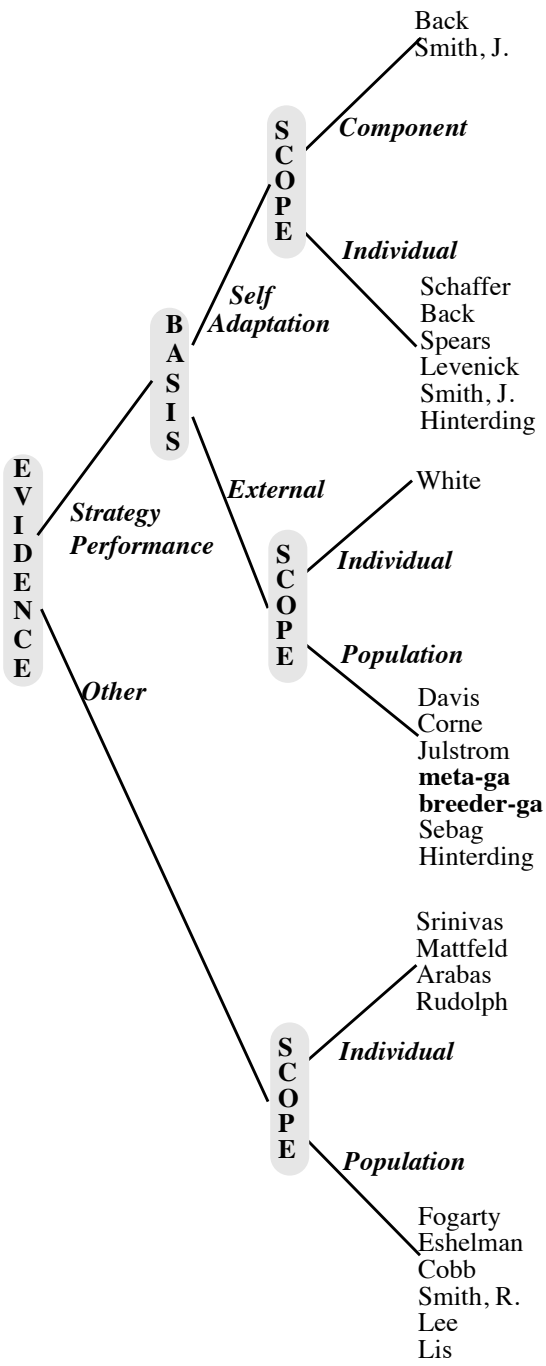
Spears, W. M. and DeJong, K. A. (1991). On the virtues of parameterized uniform crossover. Belew, R. and Booker, L. (eds), Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 230–237. Morgan Kaufmann.

Srinivas, M. and Patnaik, L. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man and Cybernetics, 24(4):656–667.

Syswerda, G. (1989). Uniform crossover in genetic algorithms. Schaffer, J. D. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, pp. 2–9. Morgan Kaufmann.

Whitley, D. and Kauth, J. (1988). Genitor: A different genetic algorithm. Proceedings of the Rocky Mountain Conference on Artificial Intelligence, pp. 118–130.

This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"



This paper has been accepted for publication in a revised form in the Issue Two of the journal "Soft Computing"